

TRSDOS/LS - DOS 6.x

"THE SOURCE"

```

LD      A,(BUFFER$+1)    ;P/u buffer hi-order addr
LD      D,A
LD      BC,13             ;Move name/ext into dest
LDIR
LD      D,(IY+9)          ;P/u dir cyl of dest
POP     BC                ;Rcvr DEC of source
PUSH    BC
LD      A,B               ;Calc dir sector for
AND     1FH              ; source SYS module
ADD     A,2
LD      E,A
LD      HL,(BUFFER$)      ;P/u buffer ptr for dest
CALL    WRSYS             ;Write the dir to dest
LD      A,18              ;Init "Dir write error
JP      NZ,EXIT3          ; and quit on bad write

```

The HIT entries were transferred prior

```

POP     BC                ;Rcvr DEC of source
PUSH    BC
LD      A,B               ;Test for SYSØ
CP      2
JP      NZ,DOFILØ        ;Bypass if not SYSØ
CALL    PMTSRC            ;Prompt source
IF      @MOD4
LD      B,16              ;Init to xfer BOOT track
LD      DE,Ø              ;Init track Ø, sector Ø
ENDIF
IF      @MOD2
LD      DE,(PROTSEC)      ;Get sysinfo sector
LD      A,D
OR      A
LD      B,5

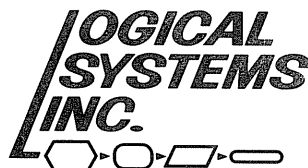
```

CONTENTS

Introduction

BACKUP/CMD -	Page 1
CLICK/FLT -	Page 71
COM/DVR -	Page 81
COMM/CMD -	Page 95
CONV/CMD -	Page 129
FLOPPY/DCT -	Page 149
FORMAT/CMD -	Page 159
FORMS/FLT -	Page 203
KSM/FLT -	Page 217
LOG/CMD -	Page 227
MEMDISK/DCT -	Page 233
PATCH/CMD -	Page 269
REPAIR/CMD -	Page 297
TAPE100/CMD -	Page 309

Appendix



8970 North 55th Street
P.O. Box 23956
Milwaukee, WI 53223

TRSDOS and TRS-80 are trademarks of Tandy Corporation
LDOS and LS-DOS are trademarks of Logical Systems, Inc.

Copyright (C) 1982, 1983, 1984 by Logical Systems, Inc.
All Rights Reserved

Introduction to Volume Three

This is volume three of three in the set of commented source code listings for LS-DOS/TRSDOS 6.2, as assembled for the TRS-80 Model 4/4P computer. This volume contains the utility programs, the standard drivers and filters, and the drive setup programs.

Each file will be preceded by a brief description of its function. A symbol table listing will follow each assembly listing.

The SVC macro file used during assembly of the utilities will be listed in Appendix A, along with the equate files generated during assembly of the resident part of the operating system.

This book should by no means be considered a tutorial on assembly language or on the workings of the LS-DOS/TRSDOS operating system. It is only the commented source code used to assemble the system utilities. It can be used for reference purposes and to view examples of interfacing outside drivers, filters and other programs to the DOS and to each other. It is not meant to replace the normal technical reference manual available from the computer manufacturer.

This product is sold on an as-is basis, and is totally unsupported by Logical Systems, Inc. No questions regarding any aspect of the source code will be answered by LSI customer or technical support. Support for LS-DOS users is provided through their OEM dealer. Support for TRSDOS 6.x is provided by Tandy Corporation. Comments or suggestions may be sent to Logical Systems, Inc. in care of the Source Code Technical Editor, but correspondence concerning these comments will not be made.

TRSDOS and TRS-80 are trademarks of Tandy Corporation
LDOS and LS-DOS are trademarks of Logical Systems, Inc.

Copyright (C) 1982, 1983, 1984 by Logical Systems, Inc.
All Rights Reserved

BACKUP/CMD - Disk and file duplication utility

Backup is assembled from three separate source files. The first contains some initialization, and the common code needed during mirror image backups and backup by class. The other modules contain the code for the two different type of backup operations.


```

00100 ;BACKUP/ASM - File/disk copy utility
0000 00110 TITLE <BACKUP - LS-DOS 6.2>
00120 ;
0000 00130 *GET BACKUP1:3
0000 00010 ;BACKUP1/ASM - Backup utility module
0000 00020 SUBTTL '<Backup initialization>'
00030 ;
0000 00040 SMALL EQU 0
0000 00050 FMT EQU 0
000A 00060 LF EQU 10
000D 00070 CR EQU 13
0060 00080 LOCK EQU 60H
00CC 00090 TKCAP EQU 0CCH
00CE 00100 PSWD EQU 0CEH
00D8 00110 DAT EQU 0D8H
00E0 00120 AUTO EQU 0E0H
1111 00130 FCNT1 EQU 1111H
1555 00140 FCNT2 EQU 1555H
42E0 00150 PASSWORD EQU 42E0H
00160 ;
0000 00170 *GET SVCMAC:3 ;SVC Macro equivalents
00180 ;SVCMAC/ASM - LS-DOS Version VI
00190 *LIST OFF
04070 *LIST ON
0000 04090 *GET COPYCOM:3 ;Copyright message
04100 ; COPYCOM - File for Copyright COMMENT block
04110 ;
0000 04120 COM '<*(C) 1982,83,84 by LSI*>'
04130 ;
2600 04140 ORG 2600H
04150 ;
04160 IF @MOD2
04170 BOOTST$ DB 03H
04180 ENDIF
04190 IF @MOD4
2600 9D 04200 BOOTST$ DB 9DH ;Boot step rate ptr
04210 ENDIF
04220 ;
04230 ; Data area
04240 ;
2601 00 04250 FTFLG$ DB 0
2602 20 04260 SPCFLD$ DC 11,' '
20 20 20 20 20 20 20
20 20
260D 00 04270 MFLG$ DB 0
260E 0000 04280 NEWPRM$ DW 0
2610 0000 04290 OLDPRM$ DW 0
2612 0000 04300 MODPRM$ DW 0
2614 0000 04310 QPARM$ DW 0
2616 0000 04320 BUFFER$ DW 0
0020 04330 FCB1$ DS 32
0020 04340 FCB2$ DS 32
0020 04350 FCB3$ DS 32
2658 04360 LILBUF$ EQU FCB3$
0008 04370 DATFLD$ DS 8
0002 04380 FMPAKD$ DS 2
0002 04390 TOPAKD$ DS 2
0001 04400 CLSFLG$ DS 1
04410 ;
04420 IF @MOD2
04430 ;

```

Backup initialization

```

                04440      SUBTTL  '<BACKUP Module - #4/2>'
                04450 ;
                04470      ENDIF
                04480 ;
                04490 ;
                04500 ;      Normal exit - no errors
                04510 ;
2685 21FC29      04520 EXIT1  LD      HL,BUCA0$      ;"Backup complete...
2688 E5          04530      PUSH     HL              ;Save msg ptr
2689 CDC926      04540      CALL     EXIT5           ;Ck if prompt for sys disk
268C E1          04550      POP      HL
268D            04560      @@DSPLY
                04570      IFEQ     00H,1
                04580      LD       HL,
                04590      ENDIF
268D 3E0A        04600      LD       A,10
268F EF          04610      RST      40
2690 182B        04620      JR       EXIT
                04630 ;
                04640 ;      Error exit
                04650 ;
2692 3E11        04660 DIRERR LD      A,17          ;Init "Dir read error
2694 01          04670      DB       1              ;Ignore next inst
2695 3E20        04680 EXIT2  LD      A,20H          ;Init illegal drive #
2697 F5          04690 EXIT3  PUSH     AF
2698 0E0D        04700      LD       C,C,CR          ;Terminate pending line
269A            04710      @@DSP
269A 3E02        04720      LD       A,2
269C EF          04730      RST      40
269D CDC926      04740      CALL     EXIT5           ;Get system disk if needed
26A0 F1          04750      POP      AF
26A1 6F          04760      LD       L,A
26A2 2600        04770      LD       H,0
26A4 F6C0        04780      OR       0C0H           ;Set short,return
26A6 4F          04790      LD       C,A           ;Error to C
26A7            04800      @@ERROR          ; for error dsply
26A7 3E1A        04810      LD       A,26
26A9 EF          04820      RST      40
26AA 180E        04830      JR       ERREXIT
                04840 ;
                04850 ;      Abort exit
                04860 ;
26AC            04870 BREAK  EQU      $
26AC 210D2A      04880 ABRTBU LD      HL,ABRTBU$      ;"Backup aborted
26AF E5          04890 EXIT4  PUSH     HL              ;Save msg ptr
26B0 CDC926      04900      CALL     EXIT5           ;Get system disk if needed
26B3 E1          04910      POP      HL
26B4            04920      @@LOGOT          ;Display the message
                04930      IFEQ     00H,1
                04940      LD       HL,
                04950      ENDIF
26B4 3E0C        04960      LD       A,12
26B6 EF          04970      RST      40
26B7 21FFFF      04980      LD       HL,-1          ;Set error return code
26BA 22C126      04990 ERREXIT LD      (RETCOD),HL
26BD            05000 EXIT     EQU      $
26BD 310000      05010 SPSAV  LD      SP,$-$        ;P/u the stack pointer
26C0 210000      05020      LD      HL,0           ;Set the return code
26C1            05030 RETCOD  EQU      $-2

```

Backup initialization

```

26C3      04900      @CKBRKC      ;Check and clear break
26C3 3E6A      00015      LD      A,106
26C5 EF      00016      RST      40
26C6      04910      @@EXIT      ;Can't return from BACKUP
26C6 3E16      00017      LD      A,22
26C8 EF      00018      RST      40
           04920 ;
           04930 ;      Get system disk if needed & zero memory used
           04940 ;
26C9      04950 EXIT5 EQU      $
26C9 110000      04960 XPARAM$ LD      DE,0      ;P/u prompt zero drive
26CC 1C      04970      INC      E      ;Ck for entry
26CD 2009      04980      JR      NZ,EXIT5A
26CF AF      04990      XOR      A
26D0 32CF27      05000      LD      (SXORD+1),A
26D3 CD0027      05010      CALL    SYSDRV$      ;Force prompt for SYSTEM
26D6 1807      05020      JR      EXIT5B
26D8 3ACF27      05030 EXIT5A LD      A,(SXORD+1)      ; else if not entered,
26DB B7      05040      OR      A      ; ck if source & dest
26DC CCFB26      05050      CALL    Z,NDSYS$      ; are same - we may need
26DF ED5B1626      05060 EXIT5B LD      DE,(BUFFER$)      ; a prompt
26E3 7A      05070      LD      A,D      ;Ck if we did a backup
26E4 B3      05080      OR      E
26E5 C8      05090      RET      Z      ;Ret if buf adr never set
26E6 210000      05100      LD      HL,0      ; else calculate how
26E9 45      05110      LD      B,L      ; many bytes in RAM
26EA      05120      @@HIGH$      ; to zero
26EA 3E64      00019      LD      A,100
26EC EF      00020      RST      40
26ED AF      05130      XOR      A
26EE ED52      05140      SBC      HL,DE      ;Get length to zero
26F0 44      05150      LD      B,H
26F1 4D      05160      LD      C,L      ; into BC
26F2 62      05170      LD      H,D      ;Pt HL to start of buffer
26F3 6B      05180      LD      L,E
26F4 13      05190      INC      DE
26F5 3600      05200      LD      (HL),0      ;Init 1st to zero
26F7 0B      05210      DEC      BC      ; & propogate it
26F8 EDB0      05220      LDIR
26FA C9      05230      RET
           05240 ;
           05250 ;      Prompt for system disk
           05260 ;
26FB 3A0E27      05270 NDSYS$ LD      A,(SRCDRV$+1)      ;On exit, if S=D <> 0
26FE B7      05280      OR      A      ; then no need to prompt
26FF C0      05290      RET      NZ
2700 3E00      05300 SYSDRV$ LD      A,0      ;P/u drive 0 indicator
2702 F620      05310      OR      20H      ;Set bit 5 for sys test
2704 E5      05320      PUSH    HL
2705 21FE28      05330      LD      HL,PMTSYS$      ;"insert system..."
2708 CDC727      05340      CALL    CURDSK
270B E1      05350      POP      HL
270C C9      05360      RET
           05370 ;
           05380 ;      Prompt source disk
           05390 ;
270D 3E00      05400 SRCDRV$ LD      A,0      ;Source drive
270F F680      05410      OR      80H      ;Set bit 7 on source
2711 E5      05420      PUSH    HL

```

Backup initialization

```

2712 211C29 05430 LD HL,PMTSRC$ ;"Insert source
2715 CDC727 05440 CALL CURDSK ;Prompt for source if needed
2718 E1 05450 POP HL
2719 C9 05460 RET
05470 ;
05480 ; Prompt source disk if needed to swap
05490 ;
271A 3AC827 05500 PMTSRC LD A,(CURDSK+1) ;P/u current drive
271D CB7F 05510 BIT 7,A ;Is source the one?
271F 20EC 05520 JR NZ,SRCDRV$ ;Jump if it is
2721 CD0D27 05530 CALL SRCDRV$ ; else prompt for it
2724 3ACF27 05540 LD A,(SXORD+1)
2727 B7 05550 OR A
2728 C0 05560 RET NZ ;Ret if source <> dest
2729 CD5E28 05570 CALL RESTOR ;Restore to cyl 0
272C C5 05580 PUSH BC
272D D5 05590 PUSH DE ;Save registers
272E E5 05600 PUSH HL
272F 21002D 05610 LD HL,BUF3$ ;Use this for I/O buffer
2732 110000 05620 LD DE,0 ;Read the BOOT
2735 CD7228 05630 CALL RDSEC
2738 E1 05640 POP HL
2739 D1 05650 POP DE ;Restore the registers
273A C1 05660 POP BC
273B C29726 05670 JP NZ,EXIT3 ;Quit on read error
273E 3A002D 05680 LD A,(BUF3$) ;P/u 1st byte of BOOT
2741 B7 05690 OR A ;If source, s/b 0
2742 2030 05700 JR NZ,PSRC3 ;Jump if not this disk
2744 C5 05710 PUSH BC
2745 D5 05720 PUSH DE
2746 E5 05730 PUSH HL
2747 FD5609 05740 LD D,(IY+9) ;P/u dir cyl
274A 1E00 05750 LD E,0 ;Pt to GAT sector
274C 21002D 05760 LD HL,BUF3$
274F CD7228 05770 CALL RDSEC ;Read the GAT
2752 FE06 05780 CP 6
2754 C29226 05790 JP NZ,DIRERR
2757 21CE2B 05800 LD HL,BUF1$+PSWD ;Ck for match with orig
275A 11CE2D 05810 LD DE,BUF3$+PSWD ; source disk
275D 060A 05820 LD B,10 ;Set match count
275F 1A 05830 PSRC1 LD A,(DE)
2760 BE 05840 CP (HL)
2761 2008 05850 JR NZ,DIFSRC ;Wrong disk if no match
2763 13 05860 INC DE ;Bump pointers
2764 23 05870 INC HL
2765 10F8 05880 DJNZ PSRC1 ;Loop for 10 compares
2767 E1 05890 POP HL ;Was a match,
2768 D1 05900 POP DE ; restore and return
2769 C1 05910 POP BC
276A C9 05920 RET
276B 05930 DIF SRC @@DSPLY DIF SRC$ ;"wake up...
00021 IFEQ 01H,1
276B 215D29 00022 LD HL,DIF SRC$
00023 ENDIF
276E 3E0A 00024 LD A,10
2770 EF 00025 RST 40
2771 E1 05940 POP HL ;Clean the stack
2772 D1 05950 POP DE
2773 C1 05960 POP BC

```

Backup initialization

```

2774 AF      05970 PSRC3 XOR A ;Show not current disk
2775 32C827 05980 LD (CURDSK+1),A
2778 18A0    05990 JR PMTSRC ;Loop to re-prompt
          06000 ;
          06010 ; Destination disk selection
          06020 ;
277A 3E00    06030 DSTDRV$ LD A,0 ;Dest drive
277C F640    06040 OR 40H ;Set dest diskette code
277E E5      06050 PUSH HL
277F 213A29 06060 LD HL,PMTDST$ ;"insert dest..."
2782 CDC727 06070 CALL CURDSK
2785 E1      06080 POP HL
2786 C9      06090 RET
          06100 ;
          06110 ; Prompt destination if needed
          06120 ;
2787 3AC827 06130 PMTDST LD A,(CURDSK+1) ;P/u current disk/drive &
278A CB77    06140 BIT 6,A ; ck if destination disk
278C 20EC    06150 JR NZ,DSTDRV$ ;Jump if it is
278E CD7A27 06160 CALL DSTDRV$ ; else request swap
2791 3ACF27 06170 LD A,(SXORD+1)
2794 B7      06180 OR A
2795 C0      06190 RET NZ ;Ret if source <> dest
2796 CD5E28 06200 CALL RESTOR ; else restore to cyl 0
2799 C5      06210 PUSH BC
279A D5      06220 PUSH DE
279B E5      06230 PUSH HL
279C 21002D 06240 LD HL,BUF3$ ;Use this for I/O buffer
279F 110000 06250 LD DE,0 ;Pt to BOOT sector
27A2 CD7228 06260 CALL RDSEC ; & read the BOOT
27A5 E1      06270 POP HL
27A6 D1      06280 POP DE
27A7 C1      06290 POP BC
27A8 C29726 06300 JP NZ,EXIT3 ;Quit on read error
27AB 3A002D 06310 LD A,(BUF3$) ;P/u 1st byte of BOOT
27AE FE76    06320 CP 76H ;Dest s/b a HALT
27B0 C8      06330 PMTDST1 RET Z
27B1 E5      06340 PUSH HL
27B2 D5      06350 PUSH DE
27B3        06360 @@DSPLY DIFDST$ ;"not same dest..."
          00026 IFEQ 01H,1
27B3 218F29 00027 LD HL,DIFDST$
          00028 ENDIF
27B6 3E0A    00029 LD A,10
27B8 EF      00030 RST 40
27B9 D1      06370 POP DE
27BA E1      06380 POP HL
27BB AF      06390 XOR A
27BC 32C827 06400 LD (CURDSK+1),A ;Show no current diskette
27BF 18C6    06410 JR PMTDST ; and prompt again
          06420 ;
          06430 ; Force a prompt of the target disk
          06440 ;
27C1 79      06450 FRCPMT LD A,C ;P/u target drive
27C2 32C827 06460 LD (CURDSK+1),A ; with code bit
27C5 180C    06470 JR FLASH
          06480 ;
          06490 ; Routine to check if flashing prompt is needed
          06500 ;

```


Backup initialization

```

27C7 FE00      06510 CURDSK CP      0      ;P/u current disk
27C9 2872      06520      JR      Z,FLSH6  ;Match with wanted disk?
27CB 32C827    06530      LD      (CURDSK+1),A ;No, update current
27CE 3EFF      06540 SXORD  LD      A,0FFH  ;0=src & dst drive same
27D0 B7        06550      OR      A
27D1 206A      06560      JR      NZ,FLSH6  ;Jump if source <> dest
                06570 ;
                06580 ;      Routine to flash the prompt
                06590 ;
27D3 C5        06600 FLASH PUSH    BC
27D4 D5        06610      PUSH    DE
27D5 E5        06620      PUSH    HL
27D6          06630      @@FLAGS      ;IY => flag table base
27D6 3E65      00031      LD      A,101
27D8 EF        00032      RST      40
27D9 0E0D      06640      LD      C,CR      ;Write a new line
27DB          06650      @@DSP
27DB 3E02      00033      LD      A,2
27DD EF        00034      RST      40
27DE 0E0F      06660      LD      C,15      ;Cursor off
27E0          06670      @@DSP
27E0 3E02      00035      LD      A,2
27E2 EF        00036      RST      40
                06680 FLASH0
27E3          06690      @@CKBRKC      ;Check and clear break
27E3 3E6A      00037      LD      A,106
27E5 EF        00038      RST      40
27E6 CD4C28    06700      CALL    RESKFLG  ;Reset Pause,Enter,Break
27E9 01FD41    06710      LD      BC,16893  ;Delay for 1/4 sec
27EC          06720      @@PAUSE
27EC 3E10      00039      LD      A,16
27EE EF        00040      RST      40
27EF FD7E0A    06730      LD      A,(IY+'K'-'A')
27F2 E605      06740      AND      4!1      ;Wait for no ENTER!BRK
27F4 20ED      06750      JR      NZ,FLASH0
27F6 CD4C28    06760      CALL    RESKFLG  ;Reset in case BREAK
27F9          06770 FLS1  @@DSPLY      ;Display the message
                00041      IFEQ    00H,1
                00042      LD      HL,
                00043      ENDIF
27F9 3E0A      00044      LD      A,10
27FB EF        00045      RST      40
27FC 015515    06780      LD      BC,FCNT2
27FF CD1428    06790      CALL    FLS2      ;Blink start
2802 0E1D      06800      LD      C,29      ;Cursor to BOL
2804          06810      @@DSP
2804 3E02      00046      LD      A,2
2806 EF        00047      RST      40
2807 0E1E      06820      LD      C,1EH      ;Cursor erase to EOL
2809          06830      @@DSP
2809 3E02      00048      LD      A,2
280B EF        00049      RST      40
280C 011111    06840      LD      BC,FCNT1  ;Wait delay count
280F CD1428    06850      CALL    FLS2      ;Wait & ck Enter or Break
2812 18E5      06860      JR      FLS1      ;Loop until Enter
                06870 FLS2
2814          06880      @@CKBRKC      ;Check for break
2814 3E6A      00050      LD      A,106
2816 EF        00051      RST      40

```

Backup initialization

```

2817 C2AC26 06890 JP NZ,BREAK ; and abort if so
281A FD7E0A 06900 LD A,(IY+'K'-'A') ;P/u KFLAG settings
281D CB57 06910 BIT 2,A ;Enter pressed?
281F 2006 06920 JR NZ,FLS4 ;Go if so
2821 0B 06930 DEC BC ;Count down
2822 78 06940 LD A,B
2823 B1 06950 OR C
2824 20EE 06960 JR NZ,FLS2 ; and loop if more time
2826 C9 06970 RET
2827 F1 06980 FLS4 POP AF ;Pop return address
2828 06990 FLS5 @@KBD ;Clear type ahead buffer
2828 3E08 00052 LD A,8
282A EF 00053 RST 40
282B 28FB 07000 JR Z,FLS5 ;Loop til no key down
282D 0E0D 07010 LD C,0DH ;Dsply a new line
282F 07020 @@DSP
282F 3E02 00054 LD A,2
2831 EF 00055 RST 40
2832 0E0E 07030 LD C,14 ;Cursor on
2834 07040 @@DSP
2834 3E02 00056 LD A,2
2836 EF 00057 RST 40
2837 CD4C28 07050 CALL RESKFLG ;Reset Break,Enter,Pause
283A E1 07060 POP HL
283B D1 07070 POP DE ;Restore registers
283C C1 07080 POP BC
283D 3AC827 07090 FLSH6 LD A,(CURDSK+1) ;P/u drive #
2840 E607 07100 AND 7 ;Strip off code bits
2842 4F 07110 LD C,A ;Drive # to C to
2843 07120 @@GTDCT ; get DCT vector
2843 3E51 00058 LD A,81
2845 EF 00059 RST 40
07130 IF @MOD4
2846 CD6328 07140 CALL RSELECT ;Get drive status in A
07150 ENDIF
07160 IF @MOD2
07170 CALL SELECT
07180 ENDIF
2849 07 07190 RLCA
284A 07 07200 RLCA
284B C9 07210 RET
284C FD7E0A 07220 RESKFLG LD A,(IY+'K'-'A') ;Reset 3-bit field
284F E6F8 07230 AND 0F8H
2851 FD770A 07240 LD (IY+'K'-'A'),A
2854 C9 07250 RET
07260 ;
07270 ; Drive disk I/O call setups
07280 ;
2855 C5 07290 TSTDRV PUSH BC
2856 AF 07300 XOR A ;Test for drive
2857 1821 07310 JR DI01
2859 C5 07320 SELECT PUSH BC
285A 3E01 07330 LD A,1 ;Select new drive
285C 181C 07340 JR DI01
285E C5 07350 RESTOR PUSH BC
285F 3E04 07360 LD A,4 ;Restore
2861 1817 07370 JR DI01
2863 C5 07380 RSELECT PUSH BC
2864 3E07 07390 LD A,7 ;Reselect

```

Backup initialization

```

2866 1812      07400 JR      DI01
2868 C5        07410 WRSEC  PUSH  BC
2869 3E0D      07420 LD      A,13      ;Write sector
286B 180D      07430 JR      DI01
286D C5        07440 WRSYS  PUSH  BC
286E 3E0E      07450 LD      A,14      ;Write protected
2870 1808      07460 JR      DI01
2872 C5        07470 RDSEC  PUSH  BC
2873 3E09      07480 LD      A,9       ;Read sector
2875 1803      07490 JR      DI01
          07500 ;
          07510 IF      @MOD2
          07520 FMTCYL  PUSH  BC      ;Save
          07530 LD      A,15      ;I/O command
          07540 JR      DI01      ;Continue
          07550 ENDIF
          07560 ;
2877 C5        07570 VERSEC  PUSH  BC
2878 3E0A      07580 LD      A,10      ;Verify sector
287A C628      07590 DI01  ADD     A,40      ;Adjust for SVC
287C 47        07600 LD      B,A       ;Save tempy
287D 3AC827    07610 LD      A,(CURDSK+1) ;Get drive number
2880 E607      07620 AND     7         ;Strip diskette type bit
2882 4F        07630 LD      C,A       ;Load up drive register
2883 78        07640 LD      A,B       ;Get back SVC #
          07650 IF      @MOD4
2884 F3        07660 DI          ;Interrupts off
          07670 ENDIF
2885 EF        07680 RST     40
          07690 IF      @MOD4
2886 FB        07700 EI          ;Interrupts on
          07710 ENDIF
2887 C1        07720 POP     BC
2888 C9        07730 RET
          07740 ;
          07750 ;      Check for correct disk
          07760 ;
2889 D5        07770 CKSWDD  PUSH  DE      ;Save DE,BC
288A C5        07780 PUSH  BC
288B 3A0E27    07790 LD      A,(SRCDRV$+1) ;Get drive
288E 21C827    07800 LD      HL,CURDSK+1
2891 4E        07810 LD      C,(HL)      ;Get current drive
2892 77        07820 LD      (HL),A     ;Make curdsk our disk
2893 2A1626    07830 LD      HL,(BUFFER$) ;I/O buffer
2896 110200    07840 LD      DE,2       ;Trk 0, sect 2
2897          07850 PROTSEC EQU  $-2
2899 CD7228    07860 CALL   RDSEC      ;Read SIS sector
289C 201C      07870 JR      NZ,EX2     ;Quit on read error
289E 2EC6      07880 LD      L,0C6H     ;Set buffer posn
28A0 3E00      07890 LD      A,$-$     ;Get original id byte
28A1          07900 SVCTR  EQU  $-1
28A2 BE        07910 CP      (HL)      ;Is it the same disk?
28A3 200F      07920 JR      NZ,EX1     ;NZ=error exit
28A5 3C        07930 INC     A
28A6 280C      07940 JR      Z,EX1
28A8 3D        07950 DEC     A      ;If id byte 0,
28A9 2809      07960 JR      Z,EX1     ; no modifying needed
28AB 3D        07970 DEC     A      ; else dec remaining
28AC 2001      07980 JR      NZ,$+3    ;If now 0, make FFH

```

Backup initialization

```

28AE 3D      07990      DEC      A
28AF 77      08000      LD      (HL),A      ;Store the new id
                08010      IF      @MOD2
                08020      LD      L,0      ;Reset buffer
                08030      ENDIF
                08040      IF      @MOD4
28B0 6A      08050      LD      L,D      ;Reset buffer
                08060      ENDIF
28B1 CD6828  08070      CALL    WRSEC      ;Put it back, ck error later
28B4 79      08080 EX1    LD      A,C
28B5 32C827  08090      LD      (CURDSK+1),A ;Restor orig drv #
28B8 C1      08100      POP     BC
28B9 D1      08110      POP     DE
28BA 211F2A  08120 EX2    LD      HL,CANTBU$ ;Go if was write error
28BD C8      08130      RET      Z
28BE C3AF26  08140      JP      EXIT4
                08150 ;
                08160 ;
                08170 ;      Message area
                08180 ;
28C1 0A      08190 DSTWP$ DB      LF,'Destination disk is write '
                44 65 73 74 69 6E 61 74
                69 6F 6E 20 64 69 73 6B
                20 69 73 20 77 72 69 74
                65 20
28DC 70      08200      DB      'protected',CR
                72 6F 74 65 63 74 65 64
                0D
28E6 49      08210 BADMPW$ DB      'Invalid master password',CR
                6E 76 61 6C 69 64 20 6D
                61 73 74 65 72 20 70 61
                73 73 77 6F 72 64 0D
28FE 1D      08220 PMTSYS$ DB      29,30,'Insert SYSTEM disk <ENTER>',3
                1E 49 6E 73 65 72 74 20
                53 59 53 54 45 4D 20 64
                69 73 6B 20 20 3C 45 4E
                54 45 52 3E 03
291C 1D      08230 PMTSRC$ DB      29,30,'Insert SOURCE disk <ENTER>',3
                1E 49 6E 73 65 72 74 20
                53 4F 55 52 43 45 20 64
                69 73 6B 20 20 3C 45 4E
                54 45 52 3E 03
293A 1D      08240 PMTDST$ DB      29,30,'Insert DESTINATION disk '
                1E 49 6E 73 65 72 74 20
                44 45 53 54 49 4E 41 54
                49 4F 4E 20 64 69 73 6B
                20 20
2955 3C      08250      DB      '<ENTER>',3
                45 4E 54 45 52 3E 03
295D 1D      08260 DIFSRC$ DB      29,30,'* A L E R T * That',27H
                1E 2A 20 41 20 4C 20 45
                20 52 20 54 20 2A 20 20
                54 68 61 74 27
2973 73      08270      DB      's not the same source disk ',CR
                20 6E 6F 74 20 74 68 65
                20 73 61 6D 65 20 73 6F
                75 72 63 65 20 64 69 73
                6B 20 0D
298F 1D      08280 DIFDST$ DB      29,30,'* A L E R T * That',27H

```

Backup initialization

1E 2A 20 41 20 4C 20 45		
20 52 20 54 20 2A 20 20		
54 68 61 74 27		
29A5 73 008290	DB	's not the same destination disk ',CR
20 6E 6F 74 20 74 68 65		
20 73 61 6D 65 20 64 65		
73 74 69 6E 61 74 69 6F		
6E 20 64 69 73 6B 20 0D		
29C6 53 008300 CCMOD\$	DB	'Source disk is write protected; '
6F 75 72 63 65 20 64 69		
73 6B 20 69 73 20 77 72		
69 74 65 20 70 72 6F 74		
65 63 74 65 64 3B 20		
29E6 4D 008310	DB	'MOD flags not updated',CR
4F 44 20 66 6C 61 67 73		
20 6E 6F 74 20 75 70 64		
61 74 65 64 0D		
29FC 0A 008320 BUCAO\$	DB	LF,'Backup complete',CR
42 61 63 6B 75 70 20 63		
6F 6D 70 6C 65 74 65 0D		
2A0D 0A 008330 ABRTBU\$	DB	LF,'Command aborted',14,CR
43 6F 6D 6D 61 6E 64 20		
61 62 6F 72 74 65 64 0E		
0D		
2A1F 43 008340 CANTBU\$	DB	'Can''t Backup - source disk write protected',LF
61 6E 27 74 20 42 61 63		
6B 75 70 20 2D 20 73 6F		
75 72 63 65 20 64 69 73		
6B 20 77 72 69 74 65 20		
70 72 6F 74 65 63 74 65		
64 0A		
2A4A 44 008350 PROT\$	DB	'Disk contains protected files ',CR
69 73 6B 20 63 6F 6E 74		
61 69 6E 73 20 70 72 6F		
74 65 63 74 65 64 20 66		
69 6C 65 73 20 0D		
2A69 008360 BUCORE\$	DEFL	\$
2B00 008370	ORG	\$<-8+1<+8
0100 008380 BUF1\$	DS	256
0100 008390 BUF2\$	DS	256
0100 008400 BUF3\$	DS	256

Backup initialization

```

08420 ;
08430 ;
08440 ;      Backup entry point
08450 ;
08460 ;
08470 BACKUP
2E00      08480      @@CKBRKC
2E00 3E6A      08490      LD      A,106
2E02 EF      08500      RST      40
2E03 2804      08510      JR      Z,BACKUPA      ;Go ahead if no break
2E05 21FFFF      08520      LD      HL,-1      ; else abort
2E08 C9      08530      RET
08520 ;
2E09 ED73BE26 08530 BACKUPA LD      (SPSAV+1),SP      ;Save current SP
2E0D E5      08540      PUSH     HL      ;Save cmdbuf
2E0E      08550      @@BREAK 0      ;Remove any BREAK vector
08560
08570      IFEQ     01H,1
2E0E 210000      08580      LD      HL,0
08590
08600      ENDIF
2E11 3E67      08610      LD      A,103
2E13 EF      08620      RST      40
2E14      08630      @@DSPLY HELLO$      ;Welcome
08640
08650      IFEQ     01H,1
2E14 21F942      08660      LD      HL,HELLO$
08670
08680      ENDIF
2E17 3E0A      08690      LD      A,10
2E19 EF      08700      RST      40
2E1A      08710      @@FLAGS      ;IY => flag table
2E1A 3E65      08720      LD      A,101
2E1C EF      08730      RST      40
2E1D CD4C28      08740      CALL    RESKFLG      ;Reset KFLAG bits
2E20 FDCB024E      08750      BIT      1,(IY+'C'-'A')      ;Check on CMNDR active
2E24 217E43      08760      LD      HL,LDOS$
2E27 C2AF26      08770      JP      NZ,EXIT4      ; and exit if so
2E2A E1      08780      POP      HL
2E2B 7E      08790      LD      A,(HL)      ;Bypass cmdline spaces
2E2C 23      08800      INC      HL
2E2D FE20      08810      CP      ' '
2E2F 28FA      08820      JR      Z,BCK1
08830 ;
08840 ;      Scan for source partial spec
08850 ;
08860 ;
08870 ;
2E31 110226      08880      LD      DE,SPCFLD$      ;Pt to filespec field
2E34 0608      08890      LD      B,8      ;Init for file name
2E36 FE2D      08900      CP      '-'      ;Exclude matches?
2E38 2005      08910      JR      NZ,BCK2      ;If '-', set flag
2E3A 320D26      08920      LD      (MFLG$),A
2E3D 7E      08930      LD      A,(HL)      ;Get next char
2E3E 23      08940      INC      HL
2E3F CDF030      08950      BCK2   CALL    PRSPEC      ;Parse possible filename
2E42 FE2F      08960      CP      '/'      ;File ext?
2E44 200A      08970      JR      NZ,BCK3
2E46 110A26      08980      LD      DE,SPCFLD$+8      ;Reposn buffer ptr
2E49 0603      08990      LD      B,3      ;Init for 3 chars
2E4B 7E      09000      LD      A,(HL)
2E4C 23      09010      INC      HL      ;Bypass the /
2E4D CDF030      09020      CALL    PRSPEC      ;Parse extension
09030 ;
09040 ;      Determine source & destination drives
09050 ;
09060 ;

```

Backup initialization

```

2E50 FE3A      08880 BCK3  CP      ':'      ;Drive number coming?
2E52 2817      08890      JR      Z,BCK4      ;Go if so
2E54 2B        08900      DEC      HL          ;Save possible parms
2E55 E5        08910      PUSH     HL
2E56          08920      @@DSPLY SRCNUM$      ;No drives enter, so
          08974      IFEQ     01H,1
2E56 21B343    08975      LD       HL,SRCNUM$
          08976      ENDIF
2E59 3E0A      08977      LD       A,10
2E5B EF        08978      RST      40
2E5C 215826    08930      LD       HL,LILBUF$      ; prompt for them
2E5F 010001    08940      LD       BC,1<8      ;1 char response
2E62          08950      @@KEYIN
2E62 3E09      08979      LD       A,9
2E64 EF        08980      RST      40
2E65 DAAC26    08960      JP       C,ABRTBU      ;Quit on Break
2E68 7E        08970      LD       A,(HL)      ;Get response. Restore
2E69 E1        08980      POP      HL          ; command buffer. Ignore
2E6A DA        08990      DB       0DAH      ; next 2 inst with JP C,
2E6B 7E        09000 BCK4  LD       A,(HL)      ;P/u source drive #
2E6C 23        09010      INC      HL          ;Bump to separator
2E6D D630      09020      SUB      '0'      ;Adj to binary
2E6F FE08      09030      CP       8          ;Error if not in
2E71 D29526    09040      JP       NC,EXIT2      ; the range <0-7>
2E74 320E27    09050      LD       (SRCDRV$+1),A ;Stuff source drive
2E77 7E        09060 BCK5  LD       A,(HL)      ;P/u char or separator
2E78 23        09070      INC      HL          ;Bump ptr
2E79 FE3A      09080      CP       ':'      ;Find dest drive?
2E7B 281F      09090      JR       Z,BCK6      ;Get drive # if :
2E7D FE30      09100      CP       30H      ; let prepositions thru
2E7F 30F6      09110      JR       NC,BCK5
2E81 FE20      09120      CP       20H      ;Or a space separator
2E83 28F2      09130      JR       Z,BCK5
2E85 2B        09140      DEC      HL          ;Save possible parms
2E86 E5        09150      PUSH     HL
2E87          09160      @@DSPLY DSTNUM$      ;Prompt for dest drive
          09081      IFEQ     01H,1
2E87 21D143    09082      LD       HL,DSTNUM$
          09083      ENDIF
2E8A 3E0A      09084      LD       A,10
2E8C EF        09085      RST      40
2E8D 215826    09170      LD       HL,LILBUF$      ;Use for keyin buffer
2E90 010001    09180      LD       BC,1<8      ;1 char only
2E93          09190      @@KEYIN
2E93 3E09      09086      LD       A,9
2E95 EF        09087      RST      40
2E96 DAAC26    09200      JP       C,ABRTBU      ;Quit on Break
2E99 7E        09210      LD       A,(HL)      ;Get response. Restore
2E9A E1        09220      POP      HL          ; buffer. Ignore next 2
2E9B DA        09230      DB       0DAH      ; inst with JP C,nn
2E9C 7E        09240 BCK6  LD       A,(HL)      ;P/u dest drive #
2E9D 23        09250      INC      HL          ;Bump line ptr
2E9E D630      09260      SUB      '0'      ;Adjust to binary
2EA0 FE08      09270      CP       8          ;Error if not in the
2EA2 D29526    09280      JP       NC,EXIT2      ; range <0-7>
2EA5 327B27    09290      LD       (DSTDRV$+1),A ;Stuff dest drive
          09300 ;
2EA8 115442    09310      LD       DE,PRMTBL$      ;P/u parm table ptr
2EAB D5        09320      PUSH     DE          ;Also in IX to check

```

Backup initialization

```

2EAC DDE1      09330 POP      IX          ; responses
2EAE           09340 @PARAM          ;Get parms if any
2EAE 3E11      00088 LD        A,17
2EB0 EF        00089 RST        40
2EB1 21A343    09350 LD        HL,PRMERR$ ;Init "parm error
2EB4 2005      09360 JR         NZ,$EX4   ;Quit on parm error
2EB6 DD7E30    09370 LD        A,(IX+DATRSP) ;Date can only be STR
2EB9 E6C0      09380 AND        VAL!SW    ;This must be string
2EBB C2AF26    09390 $EX4 JP        NZ,EXIT4 ;Quit if not
                09400 ;
                09410 ;      Check on Source = Destination
                09420 ;
2EBE 3A0E27    09430 LD        A,(SRCDRV$+1) ;P/u source drive
2EC1 217B27    09440 LD        HL,DSTDRV$+1
2EC4 AE        09450 XOR        (HL)      ;Match against dest
2EC5 32CF27    09460 LD        (SXORD+1),A ;0 if S=D, <>0 if S<>D
2EC8 200D      09470 JR         NZ,DATPRM ;Bypass if source <> dest
2ECA           09480 @FLAGS          ;Else test if <D0> proc
2ECA 3E65      00090 LD        A,101
2ECC EF        00091 RST        40
2ECD FDCB126E 09490 BIT        5,(IY+'S'-'A')
2ED1 219642    09500 LD        HL,NOINDO$ ;"can't do single...
2ED4 C2AF26    09510 JP        NZ,EXIT4 ;Abort if from <D0>
                09520 ;
                09530 ;      Check on date entries
                09540 ;
2ED7 210000    09550 DATPRM LD        HL,0 ;P/u date="from-to"
2EDA 7C        09560 LD        A,H
2EDB B5        09570 OR         L
2EDC 282F      09580 JR         Z,CKCLAS ;Bypass if not entered
2EDE 7E        09590 LD        A,(HL) ;Check for "-to"
2EDF FE2D      09600 CP         '-'
2EE1 2815      09610 JR         Z,CKTO ;Go if no From used
2EE3 3E80      09620 LD        A,80H ;Set From bit
2EE5 320126    09630 LD        (FTFLG$),A ;Note From entered
2EE8 CD0B31    09640 CALL     PAKDAT ;Pack the date entry
2EEB ED438026 09650 LD        (FMPAKD$),BC ;Save From packed date
2EEF 7E        09660 LD        A,(HL) ;Ck if more in date parm
2EF0 FE22      09670 CP         '"' ;End of string?
2EF2 280D      09680 JR         Z,FRCDAT ;Go if so
2EF4 FE2D      09690 CP         '-' ;Check for "-to"
2EF6 2015      09700 JR         NZ,CKCLAS ;Done if not
2EF8 23        09710 CKTO INC       HL ;Bypass the '-'
2EF9 7E        09720 LD        A,(HL) ;Ck for end of parm
2EFA FE22      09730 CP         '"'
2EFC 280F      09740 JR         Z,CKCLAS ;Go if done
2EFE CD0B31    09750 CALL     PAKDAT ;Pack To date
2F01 3A0126    09760 FRCDAT LD        A,(FTFLG$) ;P/u From/To flag and
2F04 F601      09770 OR         1 ; set To bit
2F06 320126    09780 LD        (FTFLG$),A
2F09 ED438226 09790 LD        (TOPAKD$),BC ;Save To packed date
                09800 ;
                09810 ;      Check on parms to force CLASS backup
                09820 ;
2F0D 0600      09830 CKCLAS LD        B,0 ;Init class flag
2F0F 110000    09840 SYSPRM LD        DE,0 ;SYS parm used?
2F12 7A        09850 LD        A,D
2F13 B3        09860 OR         E
2F14 2802      09870 JR         Z,INVPRM ;Go if not

```

Backup initialization

```

2F16 CBF0      09880      SET      6,B          ;Set 6 if SYS
2F18 110000    09890      INVPRM   LD      DE,0        ;INV parm used?
2F1B 7A        09900      LD      A,D
2F1C B3        09910      OR      E
2F1D 2802      09920      JR      Z,CKCLA1      ;Go if not
2F1F CBD8      09930      SET      3,B          ;Set 3 if INV
2F21 78        09940      CKCLA1   LD      A,B
2F22 328426    09950      LD      (CLSFLG$),A      ;Store by class flag
2F25 3A0226    09960      LD      A,(SPCFLD$)      ;Get 1st char of possible
2F28 D620      09970      SUB      ','          ; file name
2F2A 47        09980      LD      B,A          ;Save test result and
2F2B 3A0A26    09990      LD      A,(SPCFLD$+8)      ; check if extension used
2F2E D620      10000      SUB      ','          ;Ck for ext
2F30 B0        10010      OR      B          ;A <> 0 if partspec
2F31 47        10020      LD      B,A          ;Hold in reg B
                10030 ;
                10040 ;
                10050 ;
                Merge all "CLASS" parms together
2F32 DD7E0C    10060      LD      A,(IX+SYSRSP)      ;System files
2F35 DDB613    10070      OR      (IX+INVRSP)      ;Invisible files
2F38 DDB61A    10080      OR      (IX+MODRSP)      ;Mod flag files
2F3B DDB637    10090      OR      (IX+NEWRSRSP)      ;Files not on dest
2F3E DDB63E    10100      OR      (IX+OLDRSP)      ;Files on dest
2F41 DDB623    10110      OR      (IX+QRSP)        ;Query forces by class
2F44 4F        10120      LD      C,A          ;Hold value
2F45 E6A0      10130      AND      VAL!STR        ;Above parms only SWITCH
2F47 21A343    10140      LD      HL,PRMERR$      ;Init "parm error"
2F4A C2AF26    10150      JP      NZ,EXIT4        ;Quit if not switches only
2F4D B1        10160      OR      C
2F4E B0        10170      OR      B          ;Merge with partspec
2F4F DDB630    10180      OR      (IX+DATRSP)      ;D=" mm/dd/yy-mm/dd/yy"
                10190 ;
                10200 ;
                Advise backup by class if any class parameter
                10210 ;
2F52 320141    10220      LD      (CLSTST+1),A      ;Set for all flags
2F55 2806      10230      JR      Z,GETDAT        ;Z=may be mirror image
2F57           10240      @@LOGOT CLASS$          ; else log by class msg
                00092      IFEQ      01H,1
2F57 210344    00093      LD      HL,CLASS$
                00094      ENDIF
2F5A 3E0C      00095      LD      A,12
2F5C EF        00096      RST      40
                10250 ;
                10260 ;
                Recover today's date
                10270 ;
2F5D 217826    10280      GETDAT   LD      HL,DATFLD$      ;Date storage buffer
2F60           10290      @@DATE      ;Get date
2F60 3E12      00097      LD      A,18
2F62 EF        00098      RST      40
2F63 1A        10300      LD      A,(DE)          ;Check if date in system
2F64 B7        10310      OR      A
2F65 2006      10320      JR      NZ,GETGM        ;Go if it is
2F67 21EF43    10330      LD      HL,NODAT$
2F6A           10340      @@LOGOT          ;Show "no date" if none
                00099      IFEQ      00H,1
                00100      LD      HL,
                00101      ENDIF
2F6A 3E0C      00102      LD      A,12
2F6C EF        00103      RST      40

```

Backup initialization

```

2F6D D5      10350 GETGM  PUSH  DE          ;Save date$
2F6E E5      10360      PUSH  HL          ; and date buffer
2F6F 114144  10370      LD    DE,RES$    ;See if SYS modules resident
2F72         10380      @GTMOD          ; in case needed later
2F72 3E53    00104      LD    A,83
2F74 EF      00105      RST    40
2F75 2004    10390      JR     NZ,GETDAT1 ;Skip if none res'ed
2F77 ED532941 10400      LD     (RESLOC+1),DE ;Store the module loc
                10410 ;
                10420 ;      Get SYS2 loaded for password hash
                10430 ;
2F7B E1      10440 GETDAT1 POP    HL
2F7C D1      10450      POP    DE
2F7D CD6A41  10460      CALL   GETSYS2    ;Get sys2 and move date
                10470 ;
                10480 ;      Check on (X) parm for source/dest swap
                10490 ;
2F80 3A0E27  10500      LD     A,(SRCDRV$+1) ;If source is not 0,
2F83 B7      10510      OR     A          ; then let PMTSRC handle
2F84 200F    10520      JR     NZ,SRCDFT
2F86 F680    10530      OR     80H        ;Set to SRC code
2F88 4F      10540      LD     C,A        ;Save if needed
2F89 3ACA26  10550      LD     A,(XPARM$+1) ;Source is drive 0,
2F8C 3C      10560      INC     A          ; if (X), then swap
2F8D F5      10570      PUSH   AF
2F8E 211C29  10580      LD     HL,PMTSRC$
2F91 CCC127  10590      CALL   Z,FRCPMT    ;Force prompt on (X)
2F94 F1      10600      POP     AF
2F95 C40D27  10610 SRCDFT CALL   NZ,SRCDRV$    ;Prompt for source
2F98 CD5E28  10620      CALL   RESTOR     ;Get set to see if a
2F9B CDBF41  10630      CALL   CKDRV      ; source disk mounted
2F9E 280A    10640      JR     Z,GOTSRC    ;Z=ok
2FA0 F5      10650      PUSH   AF
2FA1 211C29  10660      LD     HL,PMTSRC$    ;Else prompt "Insert..."
2FA4 CDC127  10670      CALL   FRCPMT
2FA7 F1      10680      POP     AF
2FA8 18EB    10690      JR     SRCDFT      ; and then check again
                10700 ;
                10710 ;      Get source disk attributes
                10720 ;
2FAA FD7E03  10730 GOTSRC LD     A,(IY+3)    ;P/u 5" or 8" from
2FAD E620    10740      AND     20H        ; DCT+3, bit 5
2FAF 324D30  10750      LD     (TST5 8+1),A ; and save for later
2FB2 CD5528  10760      CALL   TSTDV      ;Ck for active DCT
2FB5 C29726  10770      JP     NZ,EXIT3    ; and quit if not
2FB8 21002D  10780      LD     HL,BUF3$    ;Disk buffer
                10790 ;
                10800 ;      IF @MOD2
                10810 CALL   GETPSEC    ;Get prot sector
                10820 JP     NZ,EXIT3    ;Go on error
                10830 CP     6          ;Directory?
                10840 JP     NZ,DIRERR   ;Nope, go!
                10850 ENDF
2FBB 110000  10860      LD     DE,0        ;Set to track/sector 0/0
2FBE CD7228  10870      CALL   RDSEC     ;Read boot
2FC1 C29726  10880      JP     NZ,EXIT3    ;Quit on read error
2FC4 3A022D  10890      LD     A,(BUF3$+2) ;P/u dir track
2FC7 FD7709  10900      LD     (IY+9),A    ; & stuff in table
                10910 IF @MOD2

```


Backup initialization

```

10920 LD DE,(PROTSEC) ;Get info sector
10930 ENDIF
10940 IF @MOD4
2FCA 1C 10950 INC E ;Point to SYSINFO sector
2FCB 1C 10960 INC E
10970 ENDIF
2FCC 262B 10980 LD H,BUF1$<-8 ;Use this disk buffer
2FCE CD7228 10990 CALL RDSEC ;Read the info sector
2FD1 C29726 11000 JP NZ,EXIT3 ;Quit on read error
2FD4 3AC62B 11010 LD A,(BUF1$+0C6H) ;Get & save id byte
2FD7 32A128 11020 LD (SVCTR),A
2FDA 3C 11030 INC A
2FDB 2815 11040 JR Z,CKGAT
11050 ;
11060 ; Check write protect status
11070 ;
2FDD 3D 11080 DEC A ;Need to check?
2FDE 2812 11090 JR Z,CKGAT ;Go if not
2FE0 CD5E28 11100 CALL RESTOR ;Start the drive
2FE3 CD6328 11110 CALL RSELCT ;Ck if WP
2FE6 FDB603 11120 OR (IY+3) ;Merge in soft WP
2FE9 07 11130 RLCA ;Push WP to CF
2FEA 3006 11140 JR NC,CKGAT ;Bypass if not WP
2FEC 211F2A 11150 CANTBU LD HL,CANTBU$
2FEF C3AF26 11160 JP EXIT4
11170 ;
2FF2 FD5609 11180 CKGAT LD D,(IY+9) ;Directory track,
2FF5 1E00 11190 LD E,0 ; sector 0
2FF7 21002B 11200 LD HL,BUF1$
2FFA CD7228 11210 CALL RDSEC ;Read GAT
2FFD FE06 11220 CP 6 ;Ensure directory cyl
2FFF C29226 11230 JP NZ,DIRERR ;Quit on any other error
3002 CDCF30 11240 CALL TSTMPW ;Get password if needed
11250 ;
11260 ; Check if destination formatted & not protected
11270 ;
3005 3A7B27 11280 LD A,(DSTDRTV$+1) ;If dest is not 0,
3008 B7 11290 OR A ; then let DSTDRTV handle
3009 200F 11300 JR NZ,DSTDFT
300B F640 11310 OR 40H ;Set DST code
300D 4F 11320 LD C,A ;Save if needed
300E 3ACA26 11330 LD A,(XPARM$+1) ;Dest is drive 0
3011 3C 11340 INC A ;If (X), then swap
3012 F5 11350 PUSH AF
3013 213A29 11360 LD HL,PMTDST$
3016 CCC127 11370 CALL Z,FRCPMT ;Force prompt on (X)
3019 F1 11380 POP AF
301A C47A27 11390 DSTDFT CALL NZ,DSTDRTV$ ;Get dest drive
301D CD5E28 11400 CALL RESTOR ;Restore destination
3020 CD6328 11410 CALL RSELCT ;Test it
3023 2018 11420 JR NZ,PMTDD ;Might be signal from
11430 ;HD driver
3025 07 11440 RLCA
3026 FDB603 11450 OR (IY+3) ;Merge in soft WP
3029 CB7F 11460 BIT 7,A ; Check on WP status
302B 21C128 11470 LD HL,DSTWP$ ;Dest write prot...
302E C2AF26 11480 JP NZ,EXIT4 ;Jp if write protected
11490 ;If hard drive, don't
3031 FDCB035E 11500 BIT 3,(IY+3) ; try to test for write

```

Backup initialization

```

3035 C2AE30 11510 JP NZ,RECON ; but go to re-construct
3038 CDBF41 11520 CALL CKDRV ;Ck if diskette in place
303B 280A 11530 JR Z,GOTDST
303D F5 11540 PMTDD PUSH AF ;Kludge a force of
303E 213A29 11550 LD HL,PMTDST$
3041 CDC127 11560 CALL FRCPMT
3044 F1 11570 POP AF
3045 18D3 11580 JR DSTDFT ; the destination prompt
11590 ;
11600 ; Check 5" vs 8" for forced reconstruction
11610 ;
3047 FD7E03 11620 GOTDST LD A,(IY+3)
304A E620 11630 AND 20H ;See if 5/8 mismatch
304C EE00 11640 TST5_8 XOR 0 ;P/u source size
304E C2AE30 11650 JP NZ,RECON ;Go if different
3051 110000 11660 LD DE,0
3054 CD7728 11670 CALL VERSEC ;Verify boot sector readable
3057 2806 11680 JR Z,CKDST ;Jump if ok
11690 ;
11700 ; Destination not formatted, abort
11710 ;
3059 21C942 11720 LD HL,NOFMT$ ;Init "Not formatted
305C C3AF26 11730 JP EXIT4 ;Display and abort
11740 ;
11750 ; Check destination attributes
11760 ;
305F 21002D 11770 CKDST LD HL,BUF3$
3062 110000 11780 LD DE,0 ;SET for track/sector 0/0
3065 CD7228 11790 CALL RDSEC ;Read dest boot
3068 C29726 11800 JP NZ,EXIT3
306B 3A022D 11810 LD A,(BUF3$+2) ;P/u its dir track
306E 57 11820 LD D,A ;Set up in D
306F 21002C 11830 LD HL,BUF2$
3072 5D 11840 LD E,L ; and 0 in E
3073 CD7228 11850 CALL RDSEC ;Read dest GAT
3076 FE06 11860 CP 6 ;Ensure a dir cyl
3078 C29226 11870 JP NZ,DIRERR ;Quit on any other error
307B 2ACC2B 11880 LD HL,(BUF1$+TKCAP) ;P/u source capacity
307E ED5BCC2C 11890 LD DE,(BUF2$+TKCAP) ;P/u dest capacity
3082 3AA128 11900 LD A,(SVCTR) ;If id byte was FF
3085 3C 11910 INC A
3086 2807 11920 JR Z,SHOPROT ; then force recon
3088 3D 11930 DEC A ;If id was not 0
3089 200C 11940 JR NZ,TSTCAP ; then test sizes
308B CB64 11950 BIT 4,H ;If types differ
308D 2808 11960 JR Z,TSTCAP ; force reconstruct
11970 ;
308F 11980 SHOPROT @@LOGOT PROT$ ;Show reconstruct invoked
00106 IFEQ 01H,1
308F 214A2A 00107 LD HL,PROT$
00108 ENDIF
3092 3E0C 00109 LD A,12
3094 EF 00110 RST 40
3095 1817 11990 JR RECON ;Skip next tests
12000 ;
3097 7C 12010 TSTCAP LD A,H ;Den/sides match?
3098 AA 12020 XOR D ;Force Reconstruct if
3099 E660 12030 AND 60H ; density & sides
309B 2011 12040 JR NZ,RECON ; differ

```

Backup initialization

```

309D 7D      12050      LD      A,L          ;Test # of cyls
309E 93      12060      SUB      E
309F 280A    12070      JR       Z,BYCLAS      ;Jump if same
              12080      ;
              12090      ;      Cylinder count differs - question Mirror
              12100      ;
30A1 3A0141  12110      LD      A,(CLSTST+1)    ;But don't question if
30A4 B7      12120      OR       A            ; Class parms already
30A5 C20041  12130      JP       NZ,CLSTST      ; entered
30A8 CDB730  12140      CALL     MIRROR        ;Attempt mirror?
30AB CA0041  12150      BYCLAS JP       Z,CLSTST    ;Jump if mirror to be tried
30AE        12160      RECON  @@LOGOT RECON$      ;"backup re-con...
              00111      IFEQ     01H,1
30AE 218044  00112      LD       HL,RECON$
              00113      ENDIF
30B1 3E0C    00114      LD       A,12
30B3 EF      00115      RST      40
30B4 C31341  12170      JP       MVBYCLS        ;Go do file backup
              12180      ;
              12190      ;      Different # of tracks - Prompt for mirror
              12200      ;
30B7        12210      MIRROR @@DSPLY MIRROR$      ;"Attempt mirror...
              00116      IFEQ     01H,1
30B7 219B44  00117      LD       HL,MIRROR$
              00118      ENDIF
30BA 3E0A    00119      LD       A,10
30BC EF      00120      RST      40
30BD 215926  12220      LD       HL,LILBUF$+1    ;Keyin buffer
30C0 010003  12230      QM1    LD       BC,3<8      ;3 chars max
30C3        12240      @@KEYIN
30C3 3E09    00121      LD       A,9
30C5 EF      00122      RST      40
30C6 DAAC26  12250      JP       C,ABRTBU        ;Quit on break
30C9 7E      12260      LD       A,(HL)
30CA CBAF    12270      RES      5,A            ;Convert to UC
30CC FE59    12280      CP       'Y'            ;Ret Z if Yes
30CE C9      12290      RET
              12300      ;
              12310      ;      Get & check Disk master password
              12320      ;
30CF 2ACE2B  12330      TSTMPW LD       HL,(BUF1$+PSWD) ;P/u src MPW
30D2 11E042  12340      LD       DE,PASSWORD      ;If "PASSWORD",
30D5 AF      12350      XOR       A            ; don't prompt
30D6 ED52    12360      SBC      HL,DE
30D8 C8      12370      RET       Z
30D9 110000  12380      LD       DE,$-$            ;P/u User entry
30DA        12390      MPWPRM EQU      $-2
30DC 21D344  12400      LD       HL,PMTMPW$        ;Init "Enter MPW
30DF CD6441  12410      CALL     GETMPW          ;Get the user's response
30E2 EB      12420      EX       DE,HL
30E3 2ACE2B  12430      LD       HL,(BUF1$+PSWD)
30E6 AF      12440      XOR       A
30E7 ED52    12450      SBC      HL,DE            ;Entry match?
30E9 C8      12460      RET       Z            ;Ret if MPW match
30EA 21E628  12470      LD       HL,BADMPW$        ; else init "bad MPW...
30ED C3AF26  12480      JP       EXIT4            ;Don't do the backup
              12490      ;
              12500      ;      Routine to parse partial filespecs & cvrt to UC
              12510      ;

```

Backup initialization

```

30F0 FE24      12520 PRSPEC CP      '$'          ;Wild character?
30F2 280A      12530      JR      Z,PS1          ;Always a match
30F4 FE41      12540      CP      'A'            ;Filename entered?
30F6 3006      12550      JR      NC,PS1
30F8 FE3A      12560      CP      '9'+1          ;Ck on 0-9
30FA D0        12570      RET      NC
30FB FE30      12580      CP      '0'
30FD D8        12590      RET      C
30FE FE61      12600 PS1    CP      'a'            ;Cvrt to UC if needed
3100 3802      12610      JR      C,$+4
3102 CBAF      12620      RES      5,A            ;Convert to upper case
3104 12        12630      LD      (DE),A          ;Save in partspec buffer
3105 13        12640      INC      DE             ;Bump buffer
3106 7E        12650      LD      A,(HL)          ;Get next char and
3107 23        12660      INC      HL             ; bump string ptr
3108 10E6      12670      DJNZ    PRSPEC
310A C9        12680      RET
               12690 ;
               12700 ;      Pack user date string
               12710 ;
310B 7E        12720 PAKDAT LD      A,(HL)
310C 0E2F      12730      LD      C,'/'          ;Init separator
310E CD5431    12740      CALL    PARSDAT          ;Parse entry
3111 203B      12750      JR      NZ,BADFMT        ;Jump on format error
3113 EB        12760      EX      DE,HL
3114 3A5826    12770      LD      A,(LILBUF$)      ;Is year a leap year?
3117 E603      12780      AND      3
3119 21EC44    12790      LD      HL,MAXDAYS+1     ;Set Feb to have 29 days
311C 2001      12800      JR      NZ,$+3          ; if so
311E 34        12810      INC      (HL)
311F 3A5A26    12820      LD      A,(LILBUF$+2)    ;P/u month
3122 3D        12830      DEC      A              ;Range check
3123 FE0C      12840      CP      12
3125 3027      12850      JR      NC,BADFMT        ;Go if 0 or >12
3127 2B        12860      DEC      HL             ;Point to Jan entry
3128 85        12870      ADD     A,L             ;Index the month
3129 6F        12880      LD      L,A
312A 7C        12890      LD      A,H
312B CE00      12900      ADC     A,0
312D 67        12910      LD      H,A
312E 3A5926    12920      LD      A,(LILBUF$+1)    ;P/u day entry
3131 3D        12930      DEC      A              ;Reduce for test (0->FF)
3132 BE        12940      CP      (HL)
3133 3019      12950      JR      NC,BADFMT        ;Go if too large (or 0)
3135 215A26    12960      LD      HL,LILBUF$+2     ;Pt to month
3138 7E        12970      LD      A,(HL)          ;P/u month
3139 2B        12980      DEC      HL             ;Pt to day
313A 47        12990      LD      B,A            ;Save it
313B 7E        13000      LD      A,(HL)          ;P/u day
313C 2B        13010      DEC      HL             ;Pt to year
313D 07        13020      RLCA                    ;Shift day to 3-7
313E 07        13030      RLCA
313F 07        13040      RLCA
3140 4F        13050      LD      C,A
3141 7E        13060      LD      A,(HL)          ;P/u year
3142 D650      13070      SUB      80             ;Adjust for offset
3144 3001      13080      JR      NC,$+3          ;If entry < 1980,
3146 AF        13090      XOR      A              ; then use 1980
3147 0F        13100      RRCA                    ;Shift into bits 5-7

```

Backup initialization

```

3148 0F      13110      RRCA
3149 0F      13120      RRCA
314A B0      13130      OR      B      ; & merge with month
314B 47      13140      LD      B,A
314C EB      13150      EX      DE,HL
314D C9      13160      RET
314E 21F744  13170 BADFMT LD      HL,BADFMT$
3151 C3AF26  13180      JP      EXIT4
              13190 ;
              13200 ;      Routine to parse DATE/TIME entry
              13210 ;
3154 115A26  13220 PARSDAT LD      DE,LILBUF$+2 ;Point to buf end
3157 0603    13230      LD      B,3 ;Process 3 fields
3159 D5      13240 PRSD1  PUSH   DE ;Save pointer
315A CD6931  13250      CALL  PRSD2 ;Get a digit pair
315D D1      13260      POP    DE ;Recover pointer
315E C0      13270      RET    NZ ;Ret if bad digit pair
315F 12      13280      LD      (DE),A ; else stuff the value
3160 1B      13290      DEC     DE ;Backup the pointer
3161 05      13300      DEC     B ;Loop countdown
3162 C8      13310      RET     Z
3163 7E      13320      LD      A,(HL) ;Ck for valid separator
3164 23      13330      INC     HL ;Bump pointer
3165 B9      13340      CP      C ;Separator char required
3166 28F1    13350      JR      Z,PRSD1 ;Loop if match
3168 C9      13360      RET     ;Else ret bad (NZ)
              13370 ;
              13380 ;      Routine to parse a digit pair
              13390 ;
3169 CD8031  13400 PRSD2  CALL    PRS4 ;Get a digit
316C 3010    13410      JR      NC,PRSD3 ;Jump if bad digit
316E 5F      13420      LD      E,A ;Multiply by ten
316F 07      13430      RLCA
3170 07      13440      RLCA
3171 83      13450      ADD     A,E
3172 07      13460      RLCA
3173 5F      13470      LD      E,A
3174 CD8031  13480      CALL    PRS4 ;Get another digit
3177 3005    13490      JR      NC,PRSD3 ;Jump on bad digit
3179 83      13500      ADD     A,E ;Accumulate new digit
317A 5F      13510      LD      E,A ;Save 2-digit value
317B AF      13520      XOR     A ;Clear flags
317C 7B      13530      LD      A,E ;Xfer field value
317D C9      13540      RET
317E B7      13550 PRSD3  OR      A ;Set NZ
317F C9      13560      RET
3180 7E      13570 PRS4   LD      A,(HL) ;P/u a digit &
3181 23      13580      INC     HL ;Convert to binary
3182 D630    13590      SUB     30H
3184 FE0A    13600      CP      10 ;Set CF if good
3186 C9      13610      RET
              13620 ;
              13630 ;      Save PC for later use
              13640 ;
3200      13650 CORE$  DEFL    $<-8+1<8 ;Set to next page
2E00      13660      ORG     BACKUP ;Set for MIRROR exec
3200      13670 MIRBU  EQU     CORE$
2E00      13680      LONG    MIRBU ;Load origin
              13690 ;

```


Backup initialization

2E00

13700

SUBTTL '<Mirror Image Backup>'

Mirror Image Backup

```

2E00      13720 ;
          13730 *GET      BACKUP2:3
          13740 ;BACKUP2/ASM - Mirror Image Backup
          13750 *MOD
          13760 ;
2E00 CD7A27 13770      CALL      DSTDRV$      ;Prompt for dest but
2E03 CD8727 13780      CALL      PMTDST      ; don't test yet
2E06 21002D 13790      LD        HL,BUF3$
2E09 55      13800      LD        D,L          ;Set cyl to 0
2E0A 1E01    13810      LD        E,1        ;Read sector 1 for step
2E0C CD7228 13820      CALL      RDSEC      ;Read BOOT
2E0F C29726 13830      JP        NZ,EXIT3    ;Quit on read error
2E12 3A0026 13840      LD        A,(BOOTST$) ;P/u the boot step rate
2E15 6F      13850      LD        L,A
2E16 7E      13860      LD        A,(HL)
2E17 E603    13870      AND       3          ; from bits 0-1
2E19 320630 13880      LD        (BSMIR+1),A ;Save for later
2E1C 3A022D 13890      LD        A,(BUF3$+2) ;Get dir cylinder
2E1F 57      13900      LD        D,A        ; into D
2E20 21002C 13910      LD        HL,BUF2$    ;Use this buffer now
2E23 5D      13920      LD        E,L        ;Set sector 0
2E24 CD7228 13930      CALL      RDSEC      ;Read the dest GAT
2E27 FE06    13940      CP        6          ;Expect error 6 here
2E29 3E14    13950      LD        A,20       ;Init "GAT read error
2E2B C29726 13960      JP        NZ,EXIT3    ; and abort on an error
2E2E 21CE2B 13970      LD        HL,BUF1$+0CEH ;Source GAT
2E31 11CE2C 13980      LD        DE,BUF2$+0CEH ;Dest GAT
2E34 060A    13990      LD        B,10       ;Compare pack names
2E36 1A      14000 CPRID LD        A,(DE)    ; and passwords
2E37 BE      14010      CP        (HL)
2E38 2860    14020      JR        Z,IDMATCH
          14030 ;
          14040 ;      No match - move disk name into message
          14050 ;
2E3A 21D02C 14060      LD        HL,BUF2$+0D0H
2E3D 115A32 14070      LD        DE,PACKID$+5 ;Move name into
2E40 010800 14080      LD        BC,8        ; display message field
2E43 EDB0    14090      LDIR
2E45 116932 14100      LD        DE,PACKID$+20 ;Move date into
2E48 0E08    14110      LD        C,8        ; message field
2E4A EDB0    14120      LDIR
2E4C        14130      @@LOGOT DIFID$      ;"diff pack ids..
          00123      IFEQ      01H,1
2E4C 213332 00124      LD        HL,DIFID$
          00125      ENDIF
2E4F 3E0C    00126      LD        A,12
2E51 EF      00127      RST       40
2E52        14140      @@FLAGS      ;If D0ing, don't!
2E52 3E65    00128      LD        A,101
2E54 EF      00129      RST       40
2E55 FDCB126E 14150      BIT      5,(IY+'S'-'A')
2E59 203C    14160      JR        NZ,PACKNDO ;Abort if JCL going
          14170 ;
          14180 ;      If MPW = "PASSWORD", just query Y,N
          14190 ;
2E5B 2ACE2C 14200      LD        HL,(BUF2$+0CEH) ;P/u disk MPW
2E5E 11E042 14210      LD        DE,PASSWORD ;P/u hash for "PASSWORD"
2E61 AF      14220      XOR       A
2E62 ED52    14230      SBC      HL,DE      ;Does it match disk MPW?
2E64 2818    14240      JR        Z,PMTYN    ;Go get Y or N if so

```

Mirror Image Backup

```

14250 ;
14260 ;      User must enter Current Pack's MPW to proceed
14270 ;
2E66 217232 14280 OLDMPW LD      HL,OLDMPW$      ;"What's the old MPW?
2E69 110000 14290      LD      DE,0           ;Force prompt of message
2E6C CD6441 14300      CALL    GETMPW         ;Grab user input to match
14310 ;
14320 ;      Routine to test master password for match
14330 ;
2E6F EB      14340      EX      DE,HL          ;Xfer hashed MPW to DE
2E70 2ACE2C 14350      LD      HL,(BUF2$+0CEH) ;Grab pack MPW
2E73 AF      14360      XOR      A           ;Clear carry flag
2E74 ED52    14370      SBC      HL,DE        ;Did user enter pack MPW?
2E76 21E628 14380      LD      HL,BADMPW$     ;Init "Bad MPW" just in case
2E79 C2AF26 14390      JP      NZ,EXIT4       ;Abort if no match
2E7C 1820    14400      JR      $A1          ;PW good, continue backup
14410 ;
2E7E        14420 PMTYN  @@DSPLY PMTYN$        ;"Backup anyway?"
00130      IFEQ      01H,1
2E7E 21A432 00131      LD      HL,PMTYN$
00132      ENDIF
2E81 3E0A    00133      LD      A,10
2E83 EF      00134      RST      40
2E84 215826 14430      LD      HL,LILBUF$     ;Prompt to continue
2E87 010003 14440      LD      BC,3<8        ; since ID's differ
2E8A        14450      @@KEYIN
2E8A 3E09    00135      LD      A,9
2E8C EF      00136      RST      40
2E8D DAAC26 14460      JP      C,ABRTBU       ;Exit on break
2E90 7E      14470      LD      A,(HL)
2E91 CBAF    14480      RES      5,A          ;Make answer upper case
2E93 FE59    14490      CP      'Y'          ;Was answer Yes?
2E95 2807    14500      JR      Z,$A1         ;Go if continue
2E97 C3AC26 14510 PACKNDO JP      ABRTBU       ; else abort
14520 ;
2E9A 13      14530 IDMATCH INC      DE
2E9B 23      14540      INC      HL
2E9C 1098    14550      DJNZ     CPRID
2E9E 21602C 14560 $A1 LD      HL,BUF2$+60H    ;Dest lockout table
2EA1 11602B 14570      LD      DE,BUF1$+60H    ;Source lockout table
2EA4 0660    14580      LD      B,60H          ;Init to compare 96 posns
2EA6 1A      14590 CPRLOK LD      A,(DE)       ;P/u lockout byte
2EA7 2F      14600      CPL              ;Reset all used bits
2EA8 4F      14610      LD      C,A           ; and save results
2EA9 D5      14620      PUSH     DE
2EAA 7B      14630      LD      A,E           ;Now posn to GAT byte
2EAB D660    14640      SUB      60H          ; for that track
2EAD 5F      14650      LD      E,A
2EAE 1A      14660      LD      A,(DE)       ;P/u free/used
2EAF D1      14670      POP      DE           ;Pt back to lockout
2EB0 A1      14680      AND      C           ;Merge non-locked and in use
2EB1 A6      14690      AND      (HL)        ;That much must be free on dest
2EB2 C2BC31 14700      JP      NZ,NOTMIR      ; else "dest disk flawed"
2EB5 13      14710      INC      DE
2EB6 23      14720      INC      HL
2EB7 10ED    14730      DJNZ     CPRLOK        ;Loop thru all cyls
14740 ;
14750 ;      Dest can take backup, insert HALT for swap test
14760 ;

```

Mirror Image Backup

```

2EB9 CD8727 14770 CALL PMTDST ;Prompt dest if needed
2EBC 21002D 14780 LD HL, BUF3$ ;Set up to read
2EBF 55 14790 LD D, L ; track 0,
2EC0 5D 14800 LD E, L ; sector 0
2EC1 CD7228 14810 CALL RDSEC
2EC4 C29726 14820 JP NZ, EXIT3 ;Quit on read error
2EC7 3676 14830 LD (HL), 76H ;Insert HALT to guard
2EC9 21002D 14840 LD HL, BUF3$ ; against incomplete BU
2ECC CD6828 14850 CALL WRSEC
2ECF C29726 14860 JP NZ, EXIT3 ;Quit on write error
2ED2 3A022D 14870 LD A, (BUF3$+2) ;P/U current dest/dir
2ED5 320431 14880 LD (STRDIR$+1), A ; store it for later
14890 ;
14900 ;
14910 ;
2ED8 CD1A27 14920 CALL PMTSRC ;Prompt source
2EDB FD7E09 14930 LD A, (IY+9) ;Get source dir cyl
2EDE 320F30 14940 LD (DSTDIR+1), A
14950 ;
14960 ;
14970 ;
2EE1 FD7E07 14980 LD A, (IY+7) ;P/u # of sectors per cyl
2EE4 47 14990 LD B, A ;Save # heads also
2EE5 E61F 15000 AND 1FH ;Mask all but sectors
2EE7 4F 15010 LD C, A
2EE8 0C 15020 INC C ;Adj for zero offset
2EE9 A8 15030 XOR B ;Get # of heads
2EEA 07 15040 RLCA
2EEB 07 15050 RLCA ;Shift to bits 0-2
2EEC 07 15060 RLCA
2EED 3C 15070 INC A ;Adj for 0 offset
2EEE 47 15080 LD B, A ;Init loop counter
2EEF AF 15090 XOR A ;Set sector count to 0
2EF0 81 15100 ADD A, C ;Multiply # sectors/track
2EF1 10FD 15110 DJNZ $-1 ;X # of heads/cyl
2EF3 FDCB046E 15120 BIT 5, (IY+4) ;If 2-sided diskette
2EF7 2801 15130 JR Z, $+3
2EF9 87 15140 ADD A, A ;Double the # of sectors
2EFA 327E2F 15150 LD (LDCYL4+1), A ;Save sect/cyl total
2EFD 322630 15160 LD (DUCYL5+1), A ; in many places
2F00 327B30 15170 LD (VECYL4+1), A
2F03 329031 15180 LD (RESMF6+1), A
2F06 32F130 15190 LD (RESMF2+1), A
15200 ;
15210 ;
15220 ;
2F09 47 15230 LD B, A ;Put sector count in B
2F0A 210000 15240 LD HL, 0 ;Set up to get HIGH$
2F0D C5 15250 PUSH BC ;Save the count
2F0E 45 15260 LD B, L
2F0F 15270 @@HIGH$ ;Get HIGH$
2F0F 3E64 00137 LD A, 100
2F11 EF 00138 RST 40
2F12 C1 15280 POP BC ;Recover sector count
2F13 23 15290 INC HL ;Get highest full page
2F14 25 15300 DEC H
2F15 ED5B1626 15310 LD DE, (BUFFER$) ;Get buffer addr
2F19 7C 15320 LD A, H ;Now sub buffer start
2F1A 92 15330 SUB D ; from the top

```

Mirror Image Backup

```

2F1B 0EFF      15340      LD      C,-1
2F1D 0C        15350 $A2    INC      C          ;Now count how many cyls
2F1E 90        15360      SUB      B          ; will fit in this space
2F1F 30FC      15370      JR       NC,$A2
2F21 79        15380      LD       A,C          ;This is the number of full
2F22 32892F    15390      LD       (LDCYL6+1),A    ; cylinders to move per pass
                15400 ;
                15410 ;
                15420 ;      Get source & initialize
2F25 CD1A27    15430      CALL     PMTSRC        ;Prompt source if needed
2F28 AF        15440      XOR      A          ;Init starting cylinder
2F29 32822F    15450      LD       (LDCYL5+1),A    ; to 0
2F2C 57        15460      LD       D,A          ;Set current track to 0
2F2D CD8928    15470      CALL     CKSWDD
                15480 ;
                15490 ;      Here each time a new load cycle
                15500 ;
2F30 2A1626    15510 LDTKS   LD       HL,(BUFFER$)    ;Pt to buffer start
2F33 7A        15520      LD       A,D          ;P/u cylinder to move
2F34 32A52F    15530      LD       (DUCYL+1),A    ;Save start for dump cycle
                15540 ;
                15550 ;      Here on each track loaded
                15560 ;
                15570 LDTKS1
2F37          15580      @@CKBRKC          ;Ckeck for break
2F37 3E6A      00139      LD       A,106
2F39 EF        00140      RST      40
2F3A C2AC26    15590      JP       NZ,BREAK        ; and abort if so
                15600 ;
2F3D E5        15610      PUSH     HL          ;Save buffer
2F3E 262B      15620      LD       H,BUF1$<-8    ;Pt to source GAT
2F40 6A        15630      LD       L,D          ; for this cylinder
2F41 4E        15640      LD       C,(HL)        ;P/u Free/used byte
2F42 7A        15650      LD       A,D
2F43 C660      15660      ADD      A,60H        ;Pt to Lockout byte
2F45 6F        15670      LD       L,A          ;If source track is
2F46 7E        15680      LD       A,(HL)        ; locked out, don't
2F47 2F        15690      CPL          ; back it up - BUT
2F48 A1        15700      AND      C          ; show dest is "in use"
2F49 262C      15710      LD       H,BUF2$<-8    ;Pt to dest lockout
2F4B 4E        15720      LD       C,(HL)        ;P/u dest lockout byte
2F4C B1        15730      OR       C          ;Merge with source
2F4D 6A        15740      LD       L,D          ;Xfer pattern to FREE
2F4E 77        15750      LD       (HL),A        ; field of dest
2F4F B9        15760      CP       C
2F50 E1        15770      POP      HL          ;Recover buffer
2F51 CA8D2F    15780      JP       Z,LDCYL7      ;Go if ignore this track
                15790 ;
                15800 ;      Get source disk and load
                15810 ;
2F54 CD1A27    15820      CALL     PMTSRC        ;Prompt source if needed
2F57 E5        15830      PUSH     HL          ;Save buffer
2F58 1E00      15840      LD       E,0          ;Start track at sector 0
2F5A 7A        15850      LD       A,D          ;This is the cylinder
2F5B 210132    15860      LD       HL,CYL$      ;Message posn to hold
2F5E CD9631    15870      CALL     CVTDEC        ; ASCII cyl number
2F61 D5        15880      PUSH     DE
2F62          15890      @@DSPLY LDCYL$          ;"loading cylinder..."
                00141      IFEQ     01H,1

```

Mirror Image Backup

```

2F62 21C231 00142 LD HL,LDCYL$
00143 ENDIF
2F65 3E0A 00144 LD A,10
2F67 EF 00145 RST 40
2F68 15900 @@DSPLY CYL$ ;"xx...
00146 IFEQ 01H,1
2F68 210132 00147 LD HL,CYL$
00148 ENDIF
2F6B 3E0A 00149 LD A,10
2F6D EF 00150 RST 40
2F6E D1 15910 POP DE ;Now set up to
2F6F E1 15920 POP HL ; read the cylinder
2F70 CD7228 15930 LDCYL2 CALL RDSEC ;Read a sector
2F73 2805 15940 JR Z,LDCYL3 ;Go if no error
2F75 FE06 15950 CP 6 ;Ok if error 6 (reading DIR
2F77 C29726 15960 JP NZ,EXIT3
2F7A 24 15970 LDCYL3 INC H ;Bump buffer and
2F7B 1C 15980 INC E ; sector number
2F7C 7B 15990 LD A,E
2F7D FE00 16000 LDCYL4 CP 0 ;High sector #
2F7F 20EF 16010 JR NZ,LDCYL2 ;Loop til cyl. finished
2F81 3E00 16020 LDCYL5 LD A,$-$ ;P/u current cylinder
2F83 3C 16030 INC A
2F84 32822F 16040 LD (LDCYL5+1),A ;Store next cyl
2F87 47 16050 LD B,A
2F88 3E00 16060 LDCYL6 LD A,$-$ ;P/u last for this pass
2F8A B8 16070 CP B ;See if memory full
2F8B 280E 16080 JR Z,LDCYL8 ; and go if so
2F8D 14 16090 LDCYL7 INC D ;Bump cyl to use
2F8E 7A 16100 LD A,D
2F8F FE60 16110 CP 60H ;Highest track #?
2F91 C2372F 16120 JP NZ,LDTKS1 ;If not, do another
2F94 3A822F 16130 LD A,(LDCYL5+1) ;Were any moved?
2F97 B7 16140 OR A ;Don't dump if not
2F98 CA8F30 16150 JP Z,MOVID
2F9B 3A822F 16160 LDCYL8 LD A,(LDCYL5+1) ;P/u last cyl loaded
2F9E 327F30 16170 LD (VECYL5+1),A ; & save for VERIFY
16180 ;
16190 ; Get ready to dump to destination
16200 ;
2FA1 2A1626 16210 LD HL,(BUFFER$) ;P/u start of buffer
2FA4 1600 16220 DUCYL LD D,$-$ ;Init starting cylinder
16230 ;
16240 DUCYL1
2FA6 16250 @@CKBRKC ;Check for break
2FA6 3E6A 00151 LD A,106
2FA8 EF 00152 RST 40
2FA9 C2AC26 16260 JP NZ,BREAK ; and abort if hit
16270 ;
16280 ; Start by making dest GAT bytes
16290 ;
2FAC E5 16300 PUSH HL ;Save buffer ptr
2FAD 262B 16310 LD H,BUF1$<-8 ;Pt to source GAT
2FAF 6A 16320 LD L,D ; at current cylinder
2FB0 4E 16330 LD C,(HL) ;Get the free/used byte
2FB1 7A 16340 LD A,D
2FB2 C660 16350 ADD A,60H ;P/u the lockout byte
2FB4 6F 16360 LD L,A ; for this cylinder
2FB5 7E 16370 LD A,(HL)

```

Mirror Image Backup

```

2FB6 2F      16380      CPL                ;Merge non-locked and
2FB7 A1      16390      AND C                ; in use bits
2FB8 262C    16400      LD H,BUF2$<-8        ;Pt to dest GAT
2FBA 4E      16410      LD C,(HL)            ;P/u its lockout byte
2FBB B1      16420      OR C                ;Merge in source info
2FBC 6A      16430      LD L,D                ;Store in dest free/used
2FBD 77      16440      LD (HL),A
2FBE B9      16450      CP C                ;Check if any in use
2FBF E1      16460      POP HL
2FC0 CA3030  16470      JP Z,DUCYL6          ; and go if not
2FC3 CD8727  16480      CALL PMTDST          ;Set up to write dest disk
2FC6 1E00    16490      LD E,0              ;Init to sector 0
2FC8 7A      16500      LD A,D                ;Get current cylinder
2FC9 B3      16510      OR E
2FCA E5      16520      PUSH HL              ;Save buffer ptr
2FCB 7A      16530      LD A,D
2FCC 210132  16540      LD HL,CYL$          ;"xx...
2FCF CD9631  16550      CALL CVTDEC          ;Convert cyl # to ASCII
2FD2 D5      16560      PUSH DE
2FD3         16570      @@DSPLY DUCYL$        ;"dumping cyl...
          00153      IFEQ 01H,1
2FD3 21D731  00154      LD HL,DUCYL$
          00155      ENDIF
2FD6 3E0A    00156      LD A,10
2FD8 EF      00157      RST 40
2FD9         16580      @@DSPLY CYL$          ;"xx...
          00158      IFEQ 01H,1
2FD9 210132  00159      LD HL,CYL$
          00160      ENDIF
2FDC 3E0A    00161      LD A,10
2FDE EF      00162      RST 40
2FDF D1      16590      POP DE              ;Recover cyl/sect
2FE0 E1      16600      POP HL              ; and buffer posn
2FE1 7A      16610      LD A,D                ;P/u track # & bypass
2FE2 B7      16620      OR A                ; if not cyl=0
2FE3 2028    16630      JR NZ,DUCYL2B
          16640      ;
          16650      IF @MOD2
          16660      LD A,(BACKUP0)          ;Get system flag
          16670      OR A                    ;System disk?
          16680      JR NZ,DUCYL2B          ;Yes, bypass!
          16690      ENDIF
          16700      ;
2FE5 B3      16710      OR E                ;Merge to test for sec=2
2FE6 FE02    16720      CP 2
2FE8 200D    16730      JR NZ,CKBOOT         ;If not 2, ck 1 or 0
2FEA 2EC6    16740      LD L,0C6H           ;Point to id byte
2FEC 7E      16750      LD A,(HL)
2FED 3C      16760      INC A                ;If X'FF', leave as is
2FEE 2818    16770      JR Z,SET0
2FF0 3D      16780      DEC A                ;If X'00', leave as is
2FF1 2815    16790      JR Z,SET0
2FF3 36FF    16800      LD (HL),-1          ;Set to X'FF'
2FF5 1811    16810      JR SET0
2FF7 E6FE    16820      AND CKBOOT          ;Sector 0 or 1?
2FF9 2012    16830      JR NZ,DUCYL2B      ;Go if not
2FFB B3      16840      OR E                ;If sector 0, just
2FFC 280D    16850      JR Z,DUCYL2A        ; bother with HALT
          16860      ;

```

Mirror Image Backup

```

        16870 ;      Keep the boot track step rate
        16880 ;
2FFE 3A0026 16890 LD      A,(BOOTST$) ;P/u step pointer
3001 6F     16900 LD      L,A      ; & update buffer ptr
3002 7E     16910 LD      A,(HL)   ;P/u this step byte
3003 E6FC   16920 AND     0FCH     ; & strip the step rate
3005 F600   16930 BSMIR  OR      0      ;Merge with the step
3007 77     16940 LD      (HL),A
3008 2E00   16950 SET0   LD      L,0      ;Reset buffer pointer
300A 01     16960 DB      1      ;Ignore next via LD BC,n
300B 3676   16970 DUCYL2 LD      (HL),76H ;Keep the HALT in dest
300D 7A     16980 DUCYL2B LD     A,D      ;P/u the cylinder #
300E FE00   16990 DSTDIR CP      0      ;Is this the dir cyl?
3010 2808   17000 JR      Z,DUCYL3 ;Go if it is
3012 CD6828 17010 CALL   WRSEC     ;Write non-dir sector
3015 C29726 17020 JP      NZ,EXIT3 ;Quit on write error
3018 1808   17030 JR      DUCYL4
301A CD6D28 17040 DUCYL3 CALL   WRSYS   ;Write dir sector
301D 3E12   17050 LD      A,18      ;Init "Dir write error
301F C29726 17060 JP      NZ,EXIT3 ; and leave if error
3022 24     17070 DUCYL4 INC     H      ;Advance buffer and
3023 1C     17080 INC     E      ; sector #
3024 7B     17090 LD      A,E
3025 FE00   17100 DUCYL5 CP      0      ;Reach end of cylinder?
3027 20B8   17110 JR      NZ,DUCYL2 ;Go if not
3029 3A822F 17120 LD      A,(LDCYL5+1) ;Count down one more
302C 3D     17130 DEC     A      ; cylinder dumped
302D 32822F 17140 LD      (LDCYL5+1),A
3030 14     17150 DUCYL6 INC     D      ;Bump cylinder #
3031 3A822F 17160 LD      A,(LDCYL5+1) ;Loop if still more
3034 B7     17170 OR      A      ; to dump
3035 C2A62F 17180 JP      NZ,DUCYL1
        17190 ;
        17200 ;      Prepare to verify
        17210 ;
3038 3AA52F 17220 LD      A,(DUCYL+1) ;P/u cyl # to start
303B 57     17230 LD      D,A
        17240 VECYL1
303C       17250 @CKBRKC ;Check if Break hit
303C 3E6A   00163 LD      A,106
303E EF     00164 RST      40
303F C2AC26 17260 JP      NZ,BREAK ;Abort on break
        17270 ;
3042 262B   17280 LD      H,BUF1$<-8 ;Pt to source GAT
3044 6A     17290 LD      L,D      ; at the current cylinder
3045 4E     17300 LD      C,(HL)   ;Get free/used byte
3046 7A     17310 LD      A,D
3047 C660   17320 ADD     A,60H     ;Pt to lockout byte for
3049 6F     17330 LD      L,A      ; the current cylinder
304A 7E     17340 LD      A,(HL)   ;P/u the locked out info
304B 2F     17350 CPL      ;Merge the non-locked and
304C A1     17360 AND     C      ; and the free/ used
304D 262C   17370 LD      H,BUF2$<-8 ;Pt to dest GAT
304F 4E     17380 LD      C,(HL)   ;P/u lockout for dest cyl
3050 B1     17390 OR      C      ;Merge source info
3051 6A     17400 LD      L,D      ;Pt to dest free/used
3052 77     17410 LD      (HL),A ; and store new value
3053 B9     17420 CP      C      ;See if in use
3054 CA8430 17430 JP      Z,VECYL6 ;Skip verify if not

```


Mirror Image Backup

```

3057 1E00      17440      LD      E,0          ;Init to sector 0
3059 7A        17450      LD      A,D          ;P/u cyl # for dsply
305A 210132    17460      LD      HL,CYL$      ;"xx..."
305D CD9631    17470      CALL    CVTDEC      ;Convert cyl # to ASCII
3060 D5        17480      PUSH    DE
3061           17490      @@DSPLY VECYL$      ;"verifying cyl..."
           00165      IFEQ    01H,1
3061 21EC31    00166      LD      HL,VECYL$
           00167      ENDIF
3064 3E0A      00168      LD      A,10
3066 EF        00169      RST      40
3067           17500      @@DSPLY CYL$      ;"xx..."
           00170      IFEQ    01H,1
3067 210132    00171      LD      HL,CYL$
           00172      ENDIF
306A 3E0A      00173      LD      A,10
306C EF        00174      RST      40
306D D1        17510      POP     DE          ;Recover cyl/sector
306E CD7728    17520 VECYL2 CALL    VERSEC      ;Verify a sector
3071 2805      17530      JR      Z,VECYL3    ;Go if no error
3073 FE06      17540      CP      6          ;Error 6 is OK
3075 C29726    17550      JP      NZ,EXIT3
3078 1C        17560 VECYL3 INC      E          ;Inc sector #
3079 7B        17570      LD      A,E
307A FE00      17580 VECYL4 CP      0          ;Check end of cylinder
307C 20F0      17590      JR      NZ,VECYL2    ;Loop if not
307E 3E00      17600 VECYL5 LD      A,0        ;Count down another
3080 3D        17610      DEC     A          ; cyl just verified
3081 327F30    17620      LD      (VECYL5+1),A
3084 14        17630 VECYL6 INC      D          ;Bump cyl # by 1
3085 3A7F30    17640      LD      A,(VECYL5+1) ;Loop if more cylinders
3088 B7        17650      OR      A          ; to verify, else go
3089 C23C30    17660      JP      NZ,VECYL1    ; back to "loading"
308C C3302F    17670      JP      LDTKS
           17680      ;
           17690      ;      All cylinders backed up, move ID info
           17700      ;
308F 0E0D      17710 MOVID  LD      C,CR      ;Print a newline
3091           17720      @@DSP
3091 3E02      00175      LD      A,2
3093 EF        00176      RST      40
3094 21CD2B    17730      LD      HL,BUF1$+0CDH ;Move in the pswd,name,
3097 11CD2C    17740      LD      DE,BUF2$+0CDH ; date, "AUTO" buffer,
309A 013300    17750      LD      BC,33H      ; & config byte
309D EDB0      17760      LDIR
309F 217826    17770      LD      HL,DATFLD$ ;Move in today's date
30A2 11D82C    17780      LD      DE,BUF2$+0D8H
30A5 0E08      17790      LD      C,8
30A7 EDB0      17800      LDIR
           17810      ;
           17820      ;      Get destination disk & write new GAT
           17830      ;
30A9 CD8727    17840      CALL    PMTDST      ;Set up to use dest disk
30AC 3A0F30    17850      LD      A,(DSTDIR+1) ;Get dir cyl
30AF 57        17860      LD      D,A        ;Set to track Dir,
30B0 1E00      17870      LD      E,0        ; sector 0
30B2 21002C    17880      LD      HL,BUF2$    ;Write the GAT back
30B5 CD6D28    17890      CALL    WRSYS
30B8 3E15      17900      LD      A,21      ;Init "GAT write error

```

Mirror Image Backup

```

30BA C29726 17910 JP NZ,EXIT3 ; and go if bad
30BD 21002D 17920 LD HL,BUF3$
30C0 CD7728 17930 CALL VERSEC ; else verify gat
30C3 FE06 17940 CP 6 ;Expect error 6
30C5 3E14 17950 LD A,20 ;Init "GAT read error now
30C7 C29726 17960 JP NZ,EXIT3 ; and quit if bad verify
30CA 3A0F30 17970 LD A,(DSTDIR+1) ;P/u cyl to use for dir
30CD 57 17980 LD D,A ;Set track = Dir
30CE 1E02 17990 LD E,2 ;Skip GAT and HIT
18000 ;
18010 ; Reset all mod flags on destination
18020 ;
30D0 2A1626 18030 RESMF LD HL,(BUFFER$) ;Use this for sector buffer
30D3 CD7228 18040 CALL RDSEC ;Read in dir record
30D6 FE06 18050 CP 6 ;Expect error 6
30D8 C29226 18060 JP NZ,DIRERR ;Abort on any other
30DB 2C 18070 INC L ;DIR+1 holds mod flag
30DC CBB6 18080 RESMF1 RES 6,(HL) ;Reset the flag
30DE 7D 18090 LD A,L
30DF C620 18100 ADD A,20H ;Index to next direc
30E1 6F 18110 LD L,A
30E2 30F8 18120 JR NC,RESMF1 ; and loop thru all 8
30E4 2E00 18130 LD L,0
30E6 CD6D28 18140 CALL WRSYS ;Write record back out
30E9 3E12 18150 LD A,18 ;Init "DIR write error
30EB C29726 18160 JP NZ,EXIT3
30EE 1C 18170 INC E ;Inc dir sector #
30EF 7B 18180 LD A,E
30F0 FE00 18190 RESMF2 CP $-$ ;Compare highest sect this cyl
30F2 20DC 18200 JR NZ,RESMF ;Loop until complete
18210 ;
18220 IF @MOD2
18230 LD A,(STRDIR$+1) ;Get old dir cyl
18240 LD B,A ;Pass for jump
18250 LD A,(BACKUP0) ;Get system backup flag
18260 OR A ;System disk?
18270 JR NZ,CNTBAK1 ;Yes, check if dir change
18280 ENDIF
18290 ;
18300 ; Clear the HALT inst from dest
18310 ;
30F4 21002D 18320 LD HL,BUF3$
30F7 55 18330 LD D,L ;Now read the BOOT
30F8 5D 18340 LD E,L ; on the dest disk
30F9 CD7228 18350 CALL RDSEC
30FC C29726 18360 JP NZ,EXIT3 ;Quit if couldn't be read
30FF 3600 18370 LD (HL),0 ;Clear the HALT
3101 23 18380 INC HL
3102 23 18390 INC HL ;Pt to old DIR cyl
3103 0600 18400 STRDIR$ LD B,$-$ ;P/u the old DIR cyl
3105 3A0F30 18410 LD A,(DSTDIR+1) ;Update the dir cyl
3108 77 18420 LD (HL),A ; in case it changed
3109 2B 18430 DEC HL ;Pt back to buffer start
310A 2B 18440 DEC HL
310B CD6828 18450 CALL WRSEC ;Write back the BOOT
310E CC7728 18460 CALL Z,VERSEC ; and then verify it
3111 C29726 18470 JP NZ,EXIT3 ;Go if write error
3114 1C 18480 INC E ;Point to sector 1
3115 CD7228 18490 CALL RDSEC ;Read it

```

Mirror Image Backup

```

3118 C29726 18500 JP NZ,EXIT3
311B 3A0F30 18510 LD A,(DSTDIR+1) ;Do the same thing again
311E 23 18520 INC HL
311F 23 18530 INC HL
3120 77 18540 LD (HL),A ;Store new dir cyl
3121 2B 18550 DEC HL
3122 2B 18560 DEC HL
3123 CD6828 18570 CALL WRSEC ;Write it back
3126 CC7728 18580 CALL Z,VERSEC ;Verify it if written OK
3129 C29726 18590 JP NZ,EXIT3 ;Quit if we couldn't
18600 ;
312C 262C 18610 CNTBAK1 LD H,BUF2$<-8 ;Destination GAT
312E 78 18620 LD A,B ;P/u old DIR cyl
312F C660 18630 ADD A,60H ;Point to lockout table
3131 6F 18640 LD L,A
3132 4E 18650 LD C,(HL) ;Check lockout byte
3133 68 18660 LD L,B ;Pt to GAT byte
3134 7E 18670 LD A,(HL) ;Get GAT byte
3135 B1 18680 OR C
3136 B9 18690 CP C ;Anything allocated?
3137 201E 18700 JR NZ,RESMF2B ;Bypass if yes
3139 78 18710 LD A,B ;Save cylinder
313A 21002D 18720 LD HL,BUF3$ ;Write E5's to cylinder
313D 11012D 18730 LD DE,BUF3$+1 ; to remove system DAM
3140 01FF00 18740 LD BC,255
3143 36E5 18750 LD (HL),0E5H
3145 EDB0 18760 LDIR
3147 69 18770 LD L,C ;Pt back to buf3$
3148 57 18780 LD D,A ;Set cylinder # in D
3149 59 18790 LD E,C ;Start with sector 0
314A 3A7E2F 18800 LD A,(LDCYL4+1) ;Get # of sectors
314D 47 18810 LD B,A ;Set loop counter
314E CD6828 18820 RESMF2A CALL WRSEC ;Write normal sector
3151 C29726 18830 JP NZ,EXIT3
3154 1C 18840 INC E ;Step to next sector
3155 10F7 18850 DJNZ RESMF2A
3157 CD5E28 18860 RESMF2B CALL RESTOR ;Restore to track 0
18870 ;
18880 ; Attempt to clear MOD flags of source
18890 ;
315A CD1A27 18900 CALL PMTSRC ;Set up for source disk
315D FD5609 18910 LD D,(IY+9) ;Get track = Dir
18920 ;
3160 1E02 18930 LD E,2 ;Skip GAT and HIT
3162 2A1626 18940 RESMF3 LD HL,(BUFFER$) ;Use this as sector buffer
3165 CD7228 18950 CALL RDSEC ;Read source dir sector
3168 FE06 18960 CP 6 ;Expect error 6
316A C29226 18970 JP NZ,DIRERR
316D 2C 18980 INC L ;Pt to DIR + 1
316E CBB6 18990 RESMF4 RES 6,(HL) ;Turn off mod flag
3170 7D 19000 LD A,L
3171 C620 19010 ADD A,20H ;Index to next direc
3173 6F 19020 LD L,A
3174 30F8 19030 JR NC,RESMF4 ;Loop 8 times/sector
3176 2E00 19040 LD L,0
3178 CD6D28 19050 CALL WRSYS ;Write back dir sector
317B 2810 19060 JR Z,RESMF5 ;Loop on no error
317D FE0F 19070 CP 15 ;Write protected source?
317F 3E12 19080 LD A,18 ;Init "DIR write error"

```

Mirror Image Backup

```

3181 C29726 19090 JP NZ,EXIT3 ;Exit if not WP error
3184 19100 @@LOGOT CCMOD$ ;"Can't clear mod flags
00177 IFEQ 01H,1
3184 21C629 00178 LD HL,CCMOD$
00179 ENDIF
3187 3E0C 00180 LD A,12
3189 EF 00181 RST 40
19110 IF @MOD4
318A C38526 19120 JP EXIT1 ;Backup is complete
19130 ENDIF
19140 IF @MOD2
19150 JR CKWRTK0 ;Check if write cyl 0
19160 ENDIF
318D 1C 19170 RESMF5 INC E ;Bump sector #
318E 7B 19180 LD A,E
318F FE00 19190 RESMF6 CP $-$ ;Compare highest sect this cyl
3191 20CF 19200 JR NZ,RESMF3 ;Do another sector if not
19210 IF @MOD4
3193 C38526 19220 JP EXIT1 ;Backup is complete
19230 ENDIF
19240 IF @MOD2
19250 CKWRTK0 LD A,(BACKUP0) ;Get flag
19260 OR A ;Anything?
19270 JP Z,EXIT1 ;Nope, go!
19280 CALL PMTSRC ;Prompt for source
19290 CALL READ0 ;Read cyl 0
19300 JP NZ,EXIT3 ;Go on error
19310 CALL PMTDST ;Prompt for dest drive
19320 CALL FORMAT0 ;Format cylinder 0
19330 JP NZ,EXIT3 ;Go on disk error
19340 ;
19350 ; Pass original step rate to new disk
19360 ;
19370 LD HL,(BUFFER$) ;Get I/O buffer
19380 INC HL ;Bump to step rate
19390 INC HL
19400 INC HL ;+3
19410 LD A,(BSMIR+1) ;Get step
19420 LD (HL),A ;Pass to buffer
19430 LD BC,80H ;Offset to sector 1
19440 ADD HL,BC ;Point to it
19450 LD (HL),A ;Pass to buffer
19460 CALL PMTDST ;Re-fetch DCT
19470 CALL WRITE0 ;Write cylinder 0
19480 JP NZ,EXIT3 ;Go on disk error
19490 CALL PMTDST ;Fetch DCT
19500 LD A,(DSTDIR+1) ;Get new dir cyl
19510 LD (IY+9),A ;Update DCT
19520 CALL UPGAT0 ;Update GAT table
19530 JP NZ,EXIT3 ;Go on disk error
19540 JP EXIT1 ; else program completed
19550 ENDIF
19560 ;
19570 ; Routine to convert cylinder # & message stuff
19580 ;
3196 3620 19590 CVTDEC LD (HL),' ' ;Init to leading blank
3198 0664 19600 LD B,100
319A CDA831 19610 CALL CVD1
319D 3620 19620 LD (HL),' ' ;Init to blank

```

Mirror Image Backup

```

319F 060A      19630      LD      B,10
31A1 CDA831    19640      CALL    CVD1
31A4 3630      19650      LD      (HL),'0'      ;Init to leading 0
31A6 0601      19660      LD      B,1
31A8 0E00      19670 CVD1    LD      C,0      ;Init digit counter
31AA 90        19680 CVD2    SUB     B      ;Sub 10's power until carry
31AB 3803      19690      JR      C,CVD3
31AD 0C        19700      INC     C      ; and bump count
31AE 18FA      19710      JR      CVD2
31B0 80        19720 CVD3    ADD     A,B      ;Add back last sub
31B1 F5        19730      PUSH    AF
31B2 79        19740      LD      A,C      ;Check the count
31B3 B7        19750      OR      A
31B4 2803      19760      JR      Z,CVD7      ;Ignore if 0
31B6 C630      19770      ADD     A,30H      ; else change to ASCII digit
31B8 77        19780      LD      (HL),A
31B9 F1        19790 CVD7    POP     AF
31BA 23        19800      INC     HL
31BB C9        19810      RET
                19820 ;
                19830 ;      Message area
                19840 ;
31BC 210532    19850 NOTMIR LD      HL,NOTMIR$
31BF C3AF26    19860      JP      EXIT4
31C2 1D        19870 LDCYL$ DB      29,'Reading < cylinder ',3
                52 65 61 64 69 6E 67 20
                3C 20 63 79 6C 69 6E 64
                65 72 20 03
31D7 1D        19880 DUCYL$ DB      29,'Writing > cylinder ',3
                57 72 69 74 69 6E 67 20
                3E 20 63 79 6C 69 6E 64
                65 72 20 03
31EC 1D        19890 VECYL$ DB      29,'Verifying cylinder ',3
                56 65 72 69 66 79 69 6E
                67 20 63 79 6C 69 6E 64
                65 72 20 03
3201 30        19900 CYL$    DB      '000',3
                30 30 03
3205 0A        19910 NOTMIR$ DB      LF,'Backup aborted, '
                42 61 63 6B 75 70 20 61
                62 6F 72 74 65 64 2C 20
3216 64        19920      DB      'destination not mirror-image',CR
                65 73 74 69 6E 61 74 69
                6F 6E 20 6E 6F 74 20 6D
                69 72 72 6F 72 2D 69 6D
                61 67 65 0D
3233 44        19930 DIFID$ DB      'Destination disk ID is different: '
                65 73 74 69 6E 61 74 69
                6F 6E 20 64 69 73 6B 20
                49 44 20 69 73 20 64 69
                66 66 65 72 65 6E 74 3A
                20
3255 4E        19940 PACKID$ DB      'Name=XXXXXXXX Date=mm/dd/yy',CR
                61 6D 65 3D 58 58 58 58
                58 58 58 58 20 20 44 61
                74 65 3D 6D 6D 2F 64 64
                2F 79 79 0D
3272 20        19950 OLDMPW$ DB      ' Enter its Master Password'
                20 45 6E 74 65 72 20 69

```

Mirror Image Backup

```

      74 73 20 4D 61 73 74 65
      72 20 50 61 73 73 77 6F
      72 64
328D 20      19960      DB      ' or <BREAK> to abort: ',3
      6F 72 20 3C 42 52 45 41
      4B 3E 20 74 6F 20 61 62
      6F 72 74 3A 20 03
32A4 41      19970 PMTYN$ DB      'Are you sure you want to backup to it '
      72 65 20 79 6F 75 20 73
      75 72 65 20 79 6F 75 20
      77 61 6E 74 20 74 6F 20
      62 61 63 6B 75 70 20 74
      6F 20 69 74 20
32CA 3C      19980      DB      '<Y,N> ? ',3
      59 2C 4E 3E 20 3F 20 03
      19990 ;
32D3 00      20000      DC      64,0      ;PATCH space
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      00 00 00 00 00 00 00 00
      20010 ;
0600      20020 MIRSIZ EQU      $<-8+1<8-BACKUP
      20030 ;
      20040 ;
      20050 ;      Adjust PC & load address for CLASS
      20060 ;
2E00      20070      ORG      BACKUP
      20080 ;
3800      20090 CLSBU  EQU      CORE$+MIRSIZ
2E00      20100      LONG     CLSBU
      20110 ;
2E00      20120      SUBTTL   '<Backup By Class>'

```

Backup By Class

```

2E00      20140 ;
          20150 *GET    BACKUP3:3
          20160 ;BACKUP3/ASM - Backup By Class
          20170 ;
          20180 ;      Find highest available memory page
          20190 ;
2E00 210000 20200 LD      HL,0      ;Set up to get HIGH$
2E03 45      20210 LD      B,L
2E04      20220 @@HIGH$
2E04 3E64    00182 LD      A,100
2E06 EF      00183 RST      40
2E07 23      20230 INC      HL      ;Find highest available
2E08 25      20240 DEC      H      ; memory page
2E09 7C      20250 LD      A,H
2E0A 32CE32 20260 LD      (DOFIL06+1),A ;Save for later testing
2E0D 32E332 20270 LD      (DOFIL08+1),A
2E10 32DA33 20280 LD      (LSTBUF1+1),A
2E13 3EC9    20290 LD      A,0C9H
2E15 32B027 20300 LD      (PMTDST1),A ;Ignore dest disk test
2E18 CD8727 20310 CALL    PMTDST ;Prompt dest drive
          20320 ;
          20330 ;      Calculate mamximum free space per dest disk type
          20340 ;
2E1B FD7E07 20350 LD      A,(IY+7) ;P/u # heads & sect/trk
2E1E 47      20360 LD      B,A      ;Save heads
2E1F E61F    20370 AND      1FH      ;Mask all but sectors
2E21 4F      20380 LD      C,A
2E22 0C      20390 INC      C      ;Adj for zero offset
2E23 A8      20400 XOR      B      ;Get # of heads
2E24 07      20410 RLCA
2E25 07      20420 RLCA      ; in bits 0-2
2E26 07      20430 RLCA
2E27 3C      20440 INC      A      ;Adj to 0 offset
2E28 47      20450 LD      B,A      ;Init loop counter
2E29 AF      20460 XOR      A      ;Init sector count to 0
2E2A 81      20470 ADD      A,C      ;Multiply # sectors/track
2E2B 10FD    20480 DJNZ    $-1      ; x # of heads/cyl
2E2D 6F      20490 LD      L,A
2E2E 2600    20500 LD      H,0      ;Xfer to 16-bit reg
2E30 FDCB046E 20510 BIT      5,(IY+4) ;If 2-sided diskette
2E34 2801    20520 JR      Z,$+3
2E36 29      20530 ADD      HL,HL ; double the # of sectors
2E37 FD4E06 20540 LD      C,(IY+6) ;P/u # cyls & adjust for
2E3A 0D      20550 DEC      C      ; BOOT & DIR
2E3B      20560 @@MUL16 ;Calc total records
2E3B 3E5B    00184 LD      A,91
2E3D EF      00185 RST      40
2E3E 65      20570 LD      H,L      ;Results to HL
2E3F 6F      20580 LD      L,A
2E40 225932 20590 LD      (SIZSAV+1),HL ;Save for later
          20600 ;
          20610 ;      Read the BOOT sector of dest disk
          20620 ;
2E43 110100 20630 LD      DE,1 ;Track 0, sector 1
2E46 21002C 20640 LD      HL,BUF2$ ;Disk buffer area
2E49 CD7228 20650 CALL    RDSEC ;Read the sector
2E4C C29726 20660 JP      NZ,EXIT3 ;Quit on read error
2E4F 3A0026 20670 LD      A,(BOOTST$) ;Locn of boot step rate
2E52 6F      20680 LD      L,A
2E53 7E      20690 LD      A,(HL) ;Get the step rate in

```

Backup By Class

```

2E54 E603      20700      AND      3          ; bits 0 and 1
2E56 321932    20710      LD      (BSCLS+1),A      ;Save for later
2E59 3A022C    20720      LD      A,(BUF2$+2)      ;P/u dir cyl
2E5C FD7709    20730      LD      (IY+9),A          ;Stuff into DCT
                20740 ;
                20750 ;      Check id type byte
                20760 ;
2E5F CD8928    20770      CALL     CKSWDD
                20780 ;
                20790 ;      If a system backup, then check the GAT & HIT
                20800 ;
2E62 3A6042    20810      LD      A,(PRMTBL$+SYSRSP)
2E65 B7        20820      OR      A          ;P/u SYS parm response
2E66 CA1A2F    20830      JP      Z,CLSB�5          ; and skip next if not SYS
                20840 ;
                20850 ;      If already a SYSTEM disk, don't check BOOT space
                20860 ;
                20870      IF      @MOD2
                20880      CALL    PMTDST          ;Get dest data
                20890 ;
                20900      LD      A,(IY+3)          ;Get DCT data
                20910      AND     28H          ;Bit 5/3
                20920      CP      20H          ;8" floppy?
                20930      JR      NZ,SETSYS2        ;Go if not
                20940      LD      A,(IY+4)          ;Get data
                20950      AND     50H          ;Bit 6/4
                20960      CP      40H          ;DD not alien?
                20970 SETSYS2 LD      D,0          ;Cyl 0 if not
                20980      JR      NZ,$+3          ;Go if system
                20990      INC     D          ;Sysinfo on cyl 1
                21000      ENDIF
                21010 ;
2E69 210036    21020      LD      HL,HITBUF          ;Set disk buffer
2E6C 1E02      21030      LD      E,2          ; and sector 2
                21040 ;
                21050 ;      Mod II save sysinfo sector for later check
                21060 ;
                21070      IF      @MOD2
                21080      LD      (CKPROT2),DE      ;Save cyl/sect
                21090      ENDIF
                21100 ;
                21110      IF      @MOD4
2E6E CD7228    21120      CALL    RDSEC          ;Read the sysinfo sector
2E71 C29726    21130      JP      NZ,EXIT3          ;Quit on read error
2E74 3AC036    21140      LD      A,(HITBUF+0C0H)    ;P/u & test the SYSTEM
2E77 3C        21150      INC     A          ; disk byte. If already
2E78 FD5609    21160      LD      D,(IY+9)
2E7B CACC2E    21170      JP      Z,CLSB�01          ; a system disk, bypass
                21180      ENDIF
                21190 ;
                21200      IF      @MOD2
                21210 ;
                21220      LD      D,(IY+9)          ;P/u dir cyl
                21230 ;
                21240      ENDIF
                21250 ;
2E7E 5D        21260      LD      E,L          ;Set sector 0, dir trk
2E7F CD7228    21270      CALL    RDSEC          ;Read the GAT
2E82 FE06      21280      CP      6          ;Expect error 6

```


Backup By Class

```

2E84 3E14      21290      LD      A,20      ;Init "GAT read error
2E86 C29726    21300      JP      NZ,EXIT3    ;Quit on any other error
                21310 ;
                21320
2E89 0600      21330      LD      B,0      ;Need no more
2E8B FDCB035E  21340      BIT      3,(IY+3)    ; if rigid drive
2E8F 2011      21350      JR      NZ,SETSYS    ;NZ = rigid
                21360      ENDIF
                21370 ;
                21380 ;      Check GAT byte on Mod2/12
                21390      IF      @MOD2
                21400      LD      L,0CDH
                21410      BIT      7,(HL)
                21420      LD      L,0
                21430      JP      Z,CLSB01      ;Go if system disk
                21440      ENDIF
                21450 ;
                21460 ;      If ALIEN or NOT 8" space is OK
                21470 ;
                21480      IF      @MOD2
                21490      LD      A,(CKPROT2+1)
                21500      OR      A
                21510      JR      Z,SETSYS      ;Go if not
                21520      ENDIF
                21530 ;
                21540 ;      Mod II must have track 0 fully available
                21550 ;
                21560      IF      @MOD2
                21570      LD      A,(HITBUF+60H) ;Track 0 lockout data
                21580      OR      1           ;Boot/sys allocation
                21590      CP      (HL)        ;Anything here?
                21600      JP      NZ,NOTSYS    ;Yes, cannot use!
                21610      ENDIF
                21620 ;
                21630 ;      Mod II must have 16 sectors available on cyl 1
                21640 ;
                21650      IF      @MOD2
                21660      INC      HL           ;Point to cyl 1
                21670      LD      B,3         ;2 grans SD or DD
                21680      ENDIF
                21690 ;
                21700 ;      Check to be sure additional grans needed for boot
                21710 ;      are not already allocated
                21720 ;
                21730      IF      @MOD4
2E91 0602      21740      LD      B,2         ;If 8" SDEN or DDEN, then
                21750      ENDIF
                21760 ;
2E93 FDCB036E  21770      BIT      5,(IY+3)    ; need gran 1
2E97 2002      21780      JR      NZ,$+4
2E99 0606      21790      LD      B,6         ;5" needs grans 1 & 2
2E9B 7E        21800      LD      A,(HL)      ;P/u GAT byte for BOOT
2E9C A0        21810      AND      B         ; & ck for needed space
2E9D 2048      21820      JR      NZ,NOTSYS    ;Go if no free space
2E9F 7E        21830      LD      A,(HL)      ;Reserve the GAT space
2EA0 B0        21840      OR      B
2EA1 77        21850      LD      (HL),A
                21860 ;
                21870 ;      Mod II must make force locked/used cyl 0

```

Backup By Class

```

21880 ;
21890 IF @MOD2
21900 LD A,-1 ;Init
21910 LD L,0 ;Reset to beginning
21920 LD (HL),A ;Allocate cyl 0
21930 LD L,60H ;Lockout table
21940 LD (HL),A ;Lockout cyl 0
21950 ENDIF
21960 ;
21970 ;
21980 ; Mask the config byte "data/system" disk bit
21990 ;
2EA2 2ECD 22000 SETSYS LD L,0CDH ;Point to config byte
2EA4 CBBE 22010 RES 7,(HL) ; & show system disk
2EA6 CDBF 33 22020 CALL WRGAT
22030 ;
22040 ; Adjust the allocation info for BOOT/SYS
22050 ;
2EA9 1E02 22060 CLSBU0 LD E,2 ;Read the directory
2EAB CD7228 22070 CALL RDSEC ; sector containing
2EAE FE06 22080 CP 6 ; BOOT/SYS record
2EB0 3E11 22090 LD A,17 ;Init "dir read error"
2EB2 C29726 22100 JP NZ,EXIT3
2EB5 04 22110 INC B ;Code to 7 3 1
2EB6 04 22120 INC B ;Code to 8 4 2
2EB7 CB28 22130 SRA B ;Code to 4 2 1
2EB9 CB28 22140 SRA B ;Code to 2 1 0
22150 ;
22160 IF @MOD2
22170 LD A,(CKPROT2+1)
22180 OR A
22190 JR Z,CLSBU01
22200 ENDIF
22210 ;
22220 ; Mod II must force BOOT/SYS to new cyl 1
22230 ;
22240 IF @MOD2
22250 CLSBU00 LD L,16H ;Cylinder start
22260 LD (HL),1 ;Force cyl 1
22270 ENDIF
22280 ;
2EBB 2E17 22290 LD L,17H ;Point to gran alloc
2EBD 70 22300 LD (HL),B ;Reset alloc
2EBE 2E14 22310 LD L,14H ;Point to ERN
2EC0 3610 22320 LD (HL),16 ;Update # BOOT records
2EC2 2E00 22330 LD L,0
2EC4 CD6D28 22340 CALL WRSYS ;Write dir sector back
2EC7 3E12 22350 LD A,18 ;Init "dir write error"
2EC9 C29726 22360 JP NZ,EXIT3 ;Exit if so
22370 ;
22380 ; If OLD entered No SYS file check needed
22390 ;
22400 CLSBU01
2ECC 3A1026 22410 LD A,(OLDPRM$) ;Check for OLD entered
2ECF B7 22420 OR A
2ED0 2048 22430 JR NZ,CLSBU5 ;Skip SYS setup if so
22440 ;
22450 ;
22460 ; Now check the HIT positions for /SYS files

```

Backup By Class

```

22470 ;
2ED2 CD4F34 22480 CALL HITRD ;Read in destination HIT
2ED5 C29726 22490 JP NZ,EXIT3
2ED8 119735 22500 LD DE,SYSDEC ;Pt to SYS file hash codes
2EDB EB 22510 EX DE,HL ;HIT to DE, hash tbl to HL
2EDC 0610 22520 LD B,16 ;Check 16 DEC's
2EDE 1A 22530 CLSBU1 LD A,(DE) ;If dest spare, stuff
2EDF B7 22540 OR A ; with source else
2EE0 2002 22550 JR NZ,CLSBU2 ; test for match
2EE2 7E 22560 LD A,(HL)
2EE3 12 22570 LD (DE),A
2EE4 BE 22580 CLSBU2 CP (HL) ;Dest match source?
2EE5 2806 22590 JR Z,CLSBU3 ;Continue if so
2EE7 213C35 22600 NOTSYS LD HL,NOTSYS$ ;Init"Can't make sys disk...
2EEA C3AF26 22610 JP EXIT4 ;Display and quit
2EED 1C 22620 CLSBU3 INC E ;Bump to next DEC
2EEE 23 22630 INC HL ; & our table
2EEF 3E08 22640 LD A,8 ;At midpoint?
2EF1 BB 22650 CP E
2EF2 2002 22660 JR NZ,CLSBU4 ;Skip if not
2EF4 1E20 22670 LD E,20H ;Adjust DEC row #
2EF6 10E6 22680 CLSBU4 DJNZ CLSBU1
2EF8 FD5609 22690 LD D,(IY+9) ;Ok to backup SYSTEM
2EFB 1E01 22700 LD E,1 ;Init to HIT sector
2EFD 210036 22710 LD HL,HITBUF
2F00 CD6D28 22720 CALL WRSYS ;Write back dest HIT
2F03 3E17 22730 LD A,23 ;Init "HIT write error
2F05 CC4F34 22740 CALL Z,HITRD ;Verify if write OK
2F08 C29726 22750 JP NZ,EXIT3 ;Quit on any error
22760 ;
22770 ; Set up byte 'C0' in SYSINFO sector
22780 ;
22790 IF @MOD2
22800 LD DE,(CKPROT2) ;Get sysinfo sector
22810 LD E,2 ;Force sector 2
22820 ENDIF
22830 ;
22840 IF @MOD4
2F0B 110200 22850 LD DE,02 ;P/u Mod4 SYSINFO sect
22860 ENDIF
22870 ;
22880 ; HL => to HITBUF at this point
22890 ;
2F0E CD7228 22900 CALL RDSEC ;Read the sector
2F11 2EC0 22910 LD L,0C0H ;Point to type flag
2F13 36FF 22920 LD (HL),0FFH ;Set it
2F15 2E00 22930 LD L,0 ;Reset buffer
2F17 CD6828 22940 CALL WRSEC ; Write it back
22950 ;
22960 CLSBU5
2F1A CD1A27 22970 CALL PMT SRC ;Set up for source disk
2F1D CD4F34 22980 CALL HITRD ;Read source HIT
2F20 C29726 22990 JP NZ,EXIT3
23000 ;
23010 ; Start the backup of files
23020 ;
2F23 210036 23030 LD HL,HITBUF ;Init to start of HIT
2F26 1834 23040 JR SCNH3 ;Branch to start
2F28 D2 23050 OPENIT DB 'R'!80H ;R2

```

Backup By Class

```

2F29 E1      23060 SCN HIT POP HL      ;Remove top stack entry
2F2A C1      23070 SCNH1 POP BC      ;Recover DEC posn
2F2B 2636    23080      LD H,HITBUF<-8 ;HIT buf hi-order
2F2D 68      23090      LD L,B      ; and lo-order
2F2E 7D      23100 SCNH2 LD A,L      ;Get the current DEC posn
2F2F C620    23110      ADD A,20H    ;Advance to next file on
2F31 6F      23120      LD L,A      ; this dir sector until
2F32 3028    23130      JR NC,SCNH3 ; end, then go to next
2F34 2C      23140      INC L      ; dir sector in the HIT
2F35 CB6D    23150      BIT 5,L      ;Did we go off the end?
2F37 2823    23160      JR Z,SCNH3 ; (ie from 1F to 20)
2F39 3E00    23170      LD A,0
2F3A         23180 SETBIT EQU $-1
2F3B B7      23190      OR A
2F3C 281B    23200      JR Z,TOEXIT1 ;If not, all done
2F3E CD8727  23210      CALL PMTDST ;Get dest DCT in IY
2F41 210036  23220      LD HL,HITBUF
2F44 FD5609  23230      LD D,(IY+9) ;Get dir cyl
2F47 5D      23240      LD E,L      ;Point to GAT sector
2F48 CD7228  23250      CALL RDSEC ; & read it
2F4B FE06    23260      CP 6
2F4D 3E14    23270      LD A,20      ;Init "GAT read error
2F4F C29726  23280      JP NZ,EXIT3
2F52 2ECD    23290      LD L,0CDH    ;Point to config byte
2F54 CBE6    23300      SET 4,(HL)
2F56 CDBF33  23310      CALL WRGAT
2F59 C38526  23320 TOEXIT1 JP EXIT1
23330 ;
23340 ; Continue to scan the major loop
23350 ;
2F5C 7E      23360 SCNH3 LD A,(HL) ;Is HIT entry spare?
2F5D B7      23370      OR A
2F5E 28CE    23380      JR Z,SCNH2 ;Loop back if so
2F60 7D      23390      LD A,L
2F61 E6FE    23400      AND 0FEH ;Bypass if BOOT or DIR
2F63 28C9    23410      JR Z,SCNH2
2F65 45      23420      LD B,L      ;Save DEC
2F66 C5      23430      PUSH BC
2F67 CD1A27  23440      CALL PMTSRC ;Set up for source disk
2F6A FD5609  23450      LD D,(IY+9) ;P/u DIR cyl
2F6D 78      23460      LD A,B      ;Pt to dir sector of
2F6E E61F    23470      AND 1FH ; this file
2F70 C602    23480      ADD A,2 ;Adj for GAT & HIT
2F72 5F      23490      LD E,A
2F73 21002C  23500      LD HL,BUF2$ ;Read dir sector
2F76 CD7228  23510      CALL RDSEC
2F79 FE06    23520      CP 6 ;Proper errcod?
2F7B C29226  23530      JP NZ,DIRERR
2F7E 78      23540      LD A,B      ;Pt to dir record for
2F7F E6E0    23550      AND 0E0H ; the source file
2F81 6F      23560      LD L,A
2F82 262C    23570      LD H,BUF2$<-8 ;Pt to hi-order dir buf
2F84 7E      23580      LD A,(HL) ;Ignore file if not
2F85 326D31  23590      LD (ATTRIB+1),A ; assigned in directory
2F88 CB67    23600      BIT 4,A
2F8A 2831    23610      JR Z,NODOIT
2F8C CB7F    23620      BIT 7,A ;Ignore file if FXDE
2F8E C22A2F  23630      JP NZ,SCNH1
2F91 2C      23640      INC L ;Bump to DIR+1

```

Backup By Class

```

2F92 3A1226 23650 LD A,(MODPRM$) ;Bypass if Mod parm
2F95 B7 23660 OR A ; not entered
2F96 2804 23670 JR Z,SCNH4
2F98 CB76 23680 BIT 6,(HL) ;If Mod parm and bit not set
2F9A 2821 23690 JR Z,NODOIT ; skip the file
23700 ;
2F9C CB66 23710 SCNH4 BIT 4,(HL) ;Check date not current
2F9E 2809 23720 JR Z,SCNH4A
2FA0 3AA128 23730 LD A,(SVCTR)
2FA3 B7 23740 OR A ;Was date set?
2FA4 2817 23750 JR Z,NODOIT ;Bypass if not
2FA6 3C 23760 INC A ;Is date current?
2FA7 2814 23770 JR Z,NODOIT ;Bypass if not
23780 ;
2FA9 2D 23790 SCNH4A DEC L ;DIR + 0
2FAA 3A8426 23800 LD A,(CLSFLG$) ;P/u CLASS parm byte
2FAD CB76 23810 BIT 6,(HL) ;Bypass if not SYS file
2FAF 2806 23820 JR Z,CKINV
2FB1 CB77 23830 BIT 6,A ;Ok, it is, was SYS used?
2FB3 2808 23840 JR Z,NODOIT ;Go if no SYS parm
2FB5 1809 23850 JR CKNAM ; else back it up
2FB7 CB5E 23860 CKINV BIT 3,(HL) ;Test if file is INV
2FB9 2805 23870 JR Z,CKNAM
2FBB CB5F 23880 BIT 3,A ;File is, want INV files?
2FBD CA2A2F 23890 NODOIT JP Z,SCNH1 ;Don't want invisibles
2FC0 3A0226 23900 CKNAM LD A,(SPCFLD$) ;Now test filespec match
2FC3 FE20 23910 CP ' ' ;If blank, don't bother
2FC5 2007 23920 JR NZ,CKNAM0 ; to match, take it
2FC7 3A0A26 23930 LD A,(SPCFLD$+8) ;How about the extension?
2FCA FE20 23940 CP ' '
2FCC 282C 23950 JR Z,SCNH6 ;Go if no ext either
23960 ;
23970 ; Test for a filespec match
23980 ;
2FCE E5 23990 CKNAM0 PUSH HL
2FCF 7D 24000 LD A,L
2FD0 C605 24010 ADD A,5 ;Pt to filename in dir
2FD2 6F 24020 LD L,A
2FD3 110226 24030 LD DE,SPCFLD$ ;Pt to user filespec
2FD6 060B 24040 LD B,11 ;11 char max
2FD8 1A 24050 CKNAM1 LD A,(DE) ;P/u user entry
2FD9 FE24 24060 CP '$' ;Wild card character?
2FDB 2808 24070 JR Z,CKNAM2 ;Always matches
2FDD BE 24080 CP (HL) ;Same as filespec?
2FDE 2805 24090 JR Z,CKNAM2 ;Loop if so
2FE0 FE20 24100 CP ' ' ;Ignore any further?
2FE2 C2F22F 24110 JP NZ,TSTMFLG ;If not blank, no match
2FE5 23 24120 CKNAM2 INC HL ;Match so far
2FE6 13 24130 INC DE
2FE7 10EF 24140 DJNZ CKNAM1
24150 ;
24160 ; Filespec class matches, check if NOT used
24170 ;
2FE9 3A0D26 24180 LD A,(MFLG$) ;Bypass if a match but
2FEC B7 24190 OR A ; - exclude given
2FED C2292F 24200 JP NZ,SCNHIT ;- was used, skip file
2FF0 1807 24210 JR SCNH5
24220 ;
2FF2 3A0D26 24230 TSTMFLG LD A,(MFLG$) ;Ignore if NG match &

```

Backup By Class

```

2FF5 B7      24240      OR      A      ; no exclude given
2FF6 CA292F  24250      JP      Z,SCNHIT
2FF9 E1      24260 SCNH5  POP      HL      ;Rcvr ptr to DIR+0
2FFA E5      24270 SCNH6  PUSH     HL
                24280 ;
                24290 ;      Now check if date matches
                24300 ;
2FFB 23      24310      INC      HL      ;Pt to date field
2FFC CDAE33  24320      CALL     UNPACK    ;Alter date for cpr
2FFF 3A0126  24330      LD      A,(FTFLG$)
3002 07      24340      RLCA      ;Tst From bit
3003 3010     24350      JR      NC,SCNH7
3005 7A      24360      LD      A,D      ;Ignore if date was
3006 B3      24370      OR      E      ; 00/00/00 for file
3007 CA292F  24380      JP      Z,SCNHIT
300A 2A8026  24390      LD      HL,(FMPAKD$) ;P/u user entry
300D EB      24400      EX      DE,HL
300E CD8233  24410      CALL     CPHLDE    ;HL-DE
3011 EB      24420      EX      DE,HL
3012 DA292F  24430      JP      C,SCNHIT    ;Bypass if date range bad
3015 3A0126  24440 SCNH7  LD      A,(FTFLG$)
3018 0F      24450      RRCA      ;Test TO bit
3019 300E     24460      JR      NC,MATCHES ;Go if no TOPARM else
301B 7A      24470      LD      A,D      ; ck if file is dated
301C B3      24480      OR      E
301D CA292F  24490      JP      Z,SCNHIT    ;Bypass if date was 00
3020 2A8226  24500      LD      HL,(TOPAKD$) ;P/u user's packed date
3023 CD8233  24510      CALL     CPHLDE    ;HL-DE
3026 DA292F  24520      JP      C,SCNHIT    ;Bypass if out of range
3029 E1      24530 MATCHES POP      HL
302A 7D      24540 DONAM  LD      A,L      ;Pt to start of dir rec
302B E6E0     24550      AND     0E0H
302D 6F      24560      LD      L,A      ;Make sure it's on stack
302E E5      24570      PUSH     HL
302F C605     24580      ADD      A,5      ;Pt to start of filename
3031 6F      24590      LD      L,A
3032 111826  24600      LD      DE,FCB1$    ;Move filename into fcb
3035 0608     24610      LD      B,8      ;Init 8 chars for filename
3037 7E      24620 DONAM1 LD      A,(HL)  ;P/u a char from the dir
3038 FE20     24630      CP      ' '      ;Space = end of name
303A 2805     24640      JR      Z,DONAM2
303C 12      24650      LD      (DE),A    ;Move char to FCB
303D 23      24660      INC      HL      ;Bump both ptrs
303E 13      24670      INC      DE
303F 10F6     24680      DJNZ     DONAM1    ;Loop for more
3041 7D      24690 DONAM2 LD      A,L      ;Pt to file extension
3042 80      24700      ADD      A,B      ; by adding the
3043 6F      24710      LD      L,A      ; loop remainder
3044 7E      24720      LD      A,(HL)
3045 FE20     24730      CP      ' '
3047 2810     24740      JR      Z,DONAM5    ;Bypass if none there
3049 3E2F     24750      LD      A,'/'    ; else set separator
304B 12      24760      LD      (DE),A    ; into the FCB
304C 13      24770      INC      DE
304D 0603     24780      LD      B,3      ;Now move in ext
304F 7E      24790 DONAM4 LD      A,(HL)  ;P/u ext char
3050 FE20     24800      CP      ' '      ;End if no more
3052 2805     24810      JR      Z,DONAM5
3054 12      24820      LD      (DE),A    ;Put in in the FCB

```

Backup By Class

```

3055 23      24830      INC      HL          ;Bump both ptrs
3056 13      24840      INC      DE
3057 10F6    24850      DJNZ     DONAM4      ;Loop for more
3059 3E03    24860      LD       A,3         ;Terminate with ETX
305B 12      24870      LD       (DE),A
305C D5      24880      PUSH     DE          ;Save pointer to spec end
          24890 ;
          24900 ;      Check for NEW or OLD option
          24910 ;
305D 3A1026  24920      LD       A,(OLDPRM$) ;P/u parm & merge
3060 210E26  24930      LD       HL,NEWPRM$ ; with new
3063 B6      24940      OR       (HL)       ;If neither, bypass
3064 284F    24950      JR       Z,BYPASS
3066 211826  24960      LD       HL,FCB1$    ;Save current spec
3069 115826  24970      LD       DE,FCB3$
306C 012000  24980      LD       BC,32
306F EDB0    24990      LDIR
3071 D1      25000      POP      DE          ;Recover spec end
3072 D5      25010      PUSH     DE          ; needed to add drivespec
3073 CD8833  25020      CALL     MAKSPC      ;Make it a file spec
3076 CD4634  25030      CALL     GETDST      ;Bring in the dest disk
3079 2A1626  25040      LD       HL,(BUFFER$) ;Buffer is irrelevant
307C 113826  25050      LD       DE,FCB2$    ;Pt to dest spec
307F FDE5    25060      PUSH     IY
3081         25070      @@FLAGS          ;IY => flag table base
3081 3E65     00186      LD       A,101
3083 EF      00187      RST       40
3084 FDCB12C6 25080      SET      0,(IY+'S'-'A') ;Inhibit file open bit
3088 FDE1     25090      POP      IY
308A         25100      @@OPEN           ;Attempt to open
308A 3E3B     00188      LD       A,59
308C EF      00189      RST       40
308D D1      25110      POP      DE          ;Keep stack proper
308E 2812    25120      JR       Z,CKOLD      ;If file exists, ck OLD
3090 FE19    25130      CP       25          ;File access denied?
3092 280E    25140      JR       Z,CKOLD      ; means it exists
3094 FE18    25150      CP       24          ;File not found?
3096 C2292F  25160      JP       NZ,SCNHIT    ;Ignore if not
3099 3A0E26  25170      LD       A,(NEWPRM$) ;Check if NEW requested
309C B7      25180      OR       A
309D 200A     25190      JR       NZ,GODOIT    ;Go if NEW & not found
309F C3292F  25200      JP       SCNHIT
30A2 3A1026  25210      LD       A,(OLDPRM$) ;Was found, backup old
30A5 B7      25220      OR       A           ; files this time?
30A6 CA292F  25230      JP       Z,SCNHIT      ;Ignore if not OLD
30A9 D5      25240      PUSH     DE
30AA 215826  25250      LD       HL,FCB3$    ;Recover the original
30AD 111826  25260      LD       DE,FCB1$    ; file name
30B0 012000  25270      LD       BC,32
30B3 EDB0    25280      LDIR
          25290 ;
          25300 ;      Check if prompting or not (Q parm)
          25310 ;
30B5 3A1526  25320      LD       A,(QPARM$+1) ;Query each file?
30B8 B7      25330      OR       A
30B9 CA4931  25340      JP       Z,NOPRPT      ;Not if not entered
30BC         25350      @@DSPLY QUERY      ;"backup filespec ?
          00190      IFEQ     01H,1
30BC 217534  00191      LD       HL,QUERY

```

Backup By Class

```

00192      ENDIF
30BF 3E0A    00193      LD      A,10
30C1 EF      00194      RST     40
                25360 ;
                25370 ;      Display file info for user decision
                25380 ;
30C2 D1      25390      POP     DE          ;Rcvr ptr to file buf
30C3 E1      25400      POP     HL          ;Rcvr ptr to 1st dir byte
30C4 D5      25410      PUSH    DE
30C5 23      25420      INC     HL          ;Pt to MOD bit
30C6 CB76    25430      BIT     6,(HL)      ;Test MOD flag
30C8 2808    25440      JR      Z,SCDAT1    ;Go if not set
30CA 3E20    25450      LD      A,' '      ;Put a space
30CC 12      25460      LD      (DE),A
30CD 13      25470      INC     DE
30CE 3E2B    25480      LD      A,'+'
30D0 12      25490      LD      (DE),A      ;Display '+' if MOD
30D1 13      25500      INC     DE
30D2 3E20    25510      SCDAT1 LD      A,' '      ;Write a space
30D4 12      25520      LD      (DE),A
30D5 13      25530      INC     DE
30D6 23      25540      INC     HL          ;Advance to date field
30D7 EB      25550      EX      DE,HL
30D8 367B    25560      LD      (HL),'{'      ;Stuff left brace
30DA 23      25570      INC     HL
30DB EB      25580      EX      DE,HL
30DC 7E      25590      LD      A,(HL)      ;If no date, then skip
30DD B7      25600      OR      A
30DE 283D    25610      JR      Z,SCDAT4    ;Ignore if no date saved
30E0 0F      25620      RRCA
30E1 0F      25630      RRCA
30E2 0F      25640      RRCA
30E3 E61F    25650      AND     1FH
30E5 062F    25660      LD      B,2FH      ;Convert day to decimal
30E7 04      25670      SCDAT2 INC     B      ; by counting # of 10's
30E8 D60A    25680      SUB     10      ;Sub 10 from day #
30EA 30FB    25690      JR      NC,SCDAT2
30EC C63A    25700      ADD     A,3AH      ;Cvrt lo order to ASCII
30EE F5      25710      PUSH    AF      ;Save day low order
30EF 78      25720      LD      A,B      ;Stuff day hi order
30F0 12      25730      LD      (DE),A
30F1 13      25740      INC     DE          ;Bump
30F2 F1      25750      POP     AF          ;Rcvr lo order day #
30F3 12      25760      LD      (DE),A      ;Stuff low order
30F4 13      25770      INC     DE          ;Bump pointer to msg
30F5 3E2D    25780      LD      A,'-'
30F7 12      25790      LD      (DE),A      ;Stuff '-'
30F8 13      25800      INC     DE          ;Pt to month field
30F9 E5      25810      PUSH    HL      ;Save DIR ptr
30FA F5      25820      PUSH    AF      ;Save separator char
30FB 2B      25830      DEC     HL      ;Pt to DIR+1 (month+)
30FC 7E      25840      LD      A,(HL)      ;P/u month etc
30FD E60F    25850      AND     0FH      ;Strip off flags
30FF 3D      25860      DEC     A      ;(mon-1)*3 to index
3100 4F      25870      LD      C,A      ; string conversion table
3101 07      25880      RLCA      ;X2
3102 81      25890      ADD     A,C      ;X3
3103 4F      25900      LD      C,A      ;Results to BC
3104 0600    25910      LD      B,0

```


Backup By Class

```

3106 217335 25920 LD HL,MONTBL ;Ptr to month names
3109 09 25930 ADD HL,BC ;Add offset to tbl start
310A 0E03 25940 LD C,3
310C EDB0 25950 LDIR ;Move 3-char month
310E F1 25960 POP AF
310F 12 25970 LD (DE),A
3110 13 25980 INC DE ;Advance to year field
3111 3E38 25990 LD A,'8' ;Stuff 8 of 1980
3113 12 26000 LD (DE),A
3114 13 26010 INC DE ;Bump msg ptr
3115 E1 26020 POP HL ;Rcvr DIR+2
3116 7E 26030 LD A,(HL) ;P/u year field
3117 E607 26040 AND 7 ;Remove day
3119 C630 26050 ADD A,'0' ;Cvrt to ASCII
311B 12 26060 LD (DE),A ;Stuff -> msg
311C 13 26070 INC DE
311D 3E03 26080 SCDAT4 LD A,3 ;Show etx for display
311F 12 26090 LD (DE),A
3120 26100 @@DSPLY FCB1$ ;Display filename
00195 IFEQ 01H,1
3120 211826 00196 LD HL,FCB1$
00197 ENDIF
3123 3E0A 00198 LD A,10
3125 EF 00199 RST 40
3126 26110 @@DSPLY QMARK$ ;" } ? "
00200 IFEQ 01H,1
3126 216E35 00201 LD HL,QMARK$
00202 ENDIF
3129 3E0A 00203 LD A,10
312B EF 00204 RST 40
312C 2A1626 26120 LD HL,(BUFFER$) ;Get user response
312F 010003 26130 LD BC,3<8 ;3 char max
3132 26140 @@KEYIN
3132 3E09 00205 LD A,9
3134 EF 00206 RST 40
3135 DAAC26 26150 JP C,ABRTBU ;Quit on Break
3138 7E 26160 LD A,(HL) ;Get the 1st char
3139 CBAF 26170 RES 5,A ;Strip lc if present
313B FE59 26180 CP 'Y' ;Yes means move the file
313D 2808 26190 JR Z,CPYMSG ;Go if so
26200 ;
26210 ; Accept 'C' for response to set QUERY=N
26220 ;
313F D643 26230 SUB 'C' ;Was response "C"?
3141 C2292F 26240 JP NZ,SCNHIT ;Don't backup if not
3144 321526 26250 LD (QPARM$+1),A ;Set QUERY=N
3147 E3 26260 CPYMSG EX (SP),HL ;Place dummy HL below
3148 E5 26270 PUSH HL ; FCB1$ ETX pointer
26280 ;
26290 ; Display copying file info
26300 ;
3149 26310 NOPRPT @@CKBRKC ;Ck if BREAK
3149 3E6A 00207 LD A,106
314B EF 00208 RST 40
314C C2AC26 26320 JP NZ,ABRTBU ;Quit if so
314F 26330 @@LOGOT CPYFIL$ ;"copying file..."
00209 IFEQ 01H,1
314F 216534 00210 LD HL,CPYFIL$
00211 ENDIF

```

Backup By Class

```

3152 3E0C      00212      LD      A,12
3154 EF        00213      RST      40
3155 E1        26340      POP      HL          ;Get pointer where ETX
3156 360D      26350      LD      (HL),CR      ; is & replace with CR
3158 E5        26360      PUSH     HL
3159           26370      @LOGOT FCB1$      ;Display the filespec
           00214      IFEQ     01H,1
3159 211826     00215      LD      HL,FCB1$
           00216      ENDIF
315C 3E0C      00217      LD      A,12
315E EF        00218      RST      40
315F D1        26380      POP      DE          ;Rcvr ptr to CR
3160 E1        26390      POP      HL
           26400      ;
           26410      ;      Put in the drive spec
           26420      ;
3161 CD8833     26430      DOBU      CALL     MAKSPC      ;Make the filespec
3164 C1         26440      POP      BC          ;Get DEC of source
3165 C5         26450      PUSH     BC
3166 78         26460      LD      A,B          ;Test if a SYS DEC
3167 E6D8       26470      AND      0D8H
3169 C23A32     26480      JP      NZ,DOFIL0      ;Jump if not SYS
316C 3E00       26490      ATTRIB  LD      A,0      ;P/u attribute byte
316E CB77       26500      BIT      6,A          ;Don't do if not SYS
3170 CA3A32     26510      JP      Z,DOFIL0
           26520      ;
           26530      ;      Routine to copy over SYS files
           26540      ;
3173 CD8727     26550      CALL     PMTDST      ;Prompt dest drive
3176 FD5609     26560      LD      D,(IY+9)      ;P/u dir cyl of dest
3179 78         26570      LD      A,B          ;Get DEC & calc sector
317A E61F       26580      AND      1FH
317C C602       26590      ADD      A,2          ;Adj for GAT & HIT
317E 5F         26600      LD      E,A
317F 2A1626     26610      LD      HL,(BUFFER$)      ;P/u buffer addr
3182 CD7228     26620      CALL     RDSEC      ;Read dir sect
3185 FE06       26630      CP      6          ;Proper errcod?
3187 C29226     26640      JP      NZ,DIRERR
318A 78         26650      LD      A,B          ;Pt to 1st byte of
318B E6E0       26660      AND      0E0H          ; dir record
318D 6F         26670      LD      L,A
318E CB66       26680      BIT      4,(HL)      ;Go if already assigned
3190 2019       26690      JR      NZ,DOSYS1
3192 365F       26700      LD      (HL),5FH      ;Show assigned, SYS, INV
3194 23         26710      INC      HL          ; & no access
3195 3600       26720      LD      (HL),0      ;Zero out DIR+1 to DIR+4
3197 54         26730      LD      D,H
3198 5D         26740      LD      E,L
3199 13         26750      INC      DE
319A 010300     26760      LD      BC,3
319D EDB0       26770      LDIR
319F 7D         26780      LD      A,L          ;Pt HL to DIR+16
31A0 C60C       26790      ADD      A,12
31A2 6F         26800      LD      L,A
31A3 3C         26810      INC      A
31A4 5F         26820      LD      E,A          ;Pt DE to DIR+17
31A5 36FF       26830      LD      (HL),0FFH      ;Stuff X'FF' into extent
31A7 0E0F       26840      LD      C,15          ; & pswd fields
31A9 EDB0       26850      LDIR

```

Backup By Class

```

31AB 7D      26860 DOSYS1 LD      A,L      ;Pt HL to Dir+0
31AC E6E0    26870      AND      0E0H      ; of dest
31AE CB76    26880      BIT      6,(HL)    ;Guard against writing
31B0 CAE72E  26890      JP      Z,NOTSYS    ; over a non-SYS file
31B3 C605    26900      ADD      A,5        ;Pt to name field
31B5 6F      26910      LD      L,A
31B6 5F      26920      LD      E,A        ;Pt DE to name field of
31B7 262C    26930      LD      H,BUF2$<-8  ; destination
31B9 3A1726  26940      LD      A,(BUFFER$+1) ;P/u buffer hi-order addr
31BC 57      26950      LD      D,A
31BD 010D00  26960      LD      BC,13       ;Move name/ext into dest
31C0 EDB0    26970      LDIR
31C2 FD5609  26980      LD      D,(IY+9)    ;P/u dir cyl of dest
31C5 C1      26990      POP      BC        ;Rcvr DEC of source
31C6 C5      27000      PUSH     BC
31C7 78      27010      LD      A,B        ;Calc dir sector for
31C8 E61F    27020      AND      1FH       ; source SYS module
31CA C602    27030      ADD      A,2
31CC 5F      27040      LD      E,A
31CD 2A1626  27050      LD      HL,(BUFFER$) ;P/u buffer ptr for dest
31D0 CD6D28  27060      CALL     WRSYS      ;Write the dir to dest
31D3 3E12    27070      LD      A,18       ;Init "Dir write error
31D5 C29726  27080      JP      NZ,EXIT3    ; and quit on bad write
27090 ;
27100 ;
27110 ;
The HIT entries were transferred prior
31D8 C1      27120      POP      BC        ;Rcvr DEC of source
31D9 C5      27130      PUSH     BC
31DA 78      27140      LD      A,B        ;Test for SYS0
31DB FE02    27150      CP      2
31DD C23A32  27160      JP      NZ,DOFIL0   ;Bypass if not SYS0
31E0 CD1A27  27170      CALL     PMTSRC     ;Prompt source
27180 IF
31E3 0610    27190      LD      B,16       ;Init to xfer BOOT track
31E5 110000  27200      LD      DE,0        ;Init track 0, sector 0
27210 ENDF
27220 IF @MOD2
27230 LD      DE,(PROTSEC) ;Get sysinfo sector
27240 LD      A,D
27250 OR      A
27260 LD      B,5
27270 JR      Z,NBTSEC2
27280 LD      B,16
27290 LD      E,0
27300 NBTSEC2 ENDF
27310 ;
31E8 2A1626  27320      LD      HL,(BUFFER$) ;Set disk buffer
31EB CD7228  27330 RDBOOT CALL     RDSEC     ;Read sector and
31EE C29726  27340      JP      NZ,EXIT3    ; quit on error
31F1 24      27350      INC      H        ;Pt to next block
31F2 1C      27360      INC      E        ;Point to next sector
31F3 10F6    27370      DJNZ     RDBOOT    ;Continue reading boot
27380 ;
27390 ;
27400 ;
Turn off CONFIG on destination disk
31F5 2A1626  27410      LD      HL,(BUFFER$) ;Start cyl image
31F8 110102  27420      LD      DE,100H*2+1 ;Offset to sector 2 +1
31FB 19      27430      ADD      HL,DE     ;HL => config byte
31FC 36C9    27440      LD      (HL),0C9H    ;Config off

```

Backup By Class

```

27450 ;
31FE CD8727 27460 DOSYS2 CALL PMTDST ;Prompt destination
27470 IF @MOD4
3201 0610 27480 LD B,16 ;Sector count for boot
3203 110000 27490 LD DE,0 ;Init track and sector 0
27500 ENDIF
27510 IF @MOD2
27520 LD DE,(CKPROT2) ;Get dest cyl number
27530 LD A,(PROTSEC+1)
27540 LD B,5 ;Default 5 sectors
27550 OR A
27560 JR Z,NBTSECS
27570 AND D
27580 JR Z,NBTSECS
27590 LD B,16 ;Use 16 sectors
27600 NBTSECS LD E,0
27610 ENDIF
3206 2A1626 27620 LD HL,(BUFFER$) ;P/u buffer start
3209 7B 27630 WRBOOT LD A,E ;If sector 0 or 1,
320A FE02 27640 CP 2 ; correct DIRCYL &
320C 3015 27650 JR NC,WRBOOT2 ; BOOT step rate
320E B7 27660 OR A
320F 280A 27670 JR Z,WRBOOT1 ;If sec 0 only dir cyl
27680 ;
3211 3A0026 27690 LD A,(BOOTST$) ;P/u step pointer
3214 6F 27700 LD L,A
3215 7E 27710 LD A,(HL) ;P/u BOOT step rate
3216 E6FC 27720 AND 0FCH ;Strip the rate
3218 F600 27730 BSCLS OR 0 ;Merge dest rate
321A 77 27740 LD (HL),A
321B FD7E09 27750 WRBOOT1 LD A,(IY+9) ;P/u DIR cyl
321E 2E02 27760 LD L,2
3220 77 27770 LD (HL),A
3221 2E00 27780 LD L,0 ;Restart to buf start
3223 CD6828 27790 WRBOOT2 CALL WRSEC ;Write dest boot sector
3226 C29726 27800 JP NZ,EXIT3 ;Quit on error
3229 24 27810 INC H ;Bump buffer page
322A 1C 27820 INC E ;Bump sector
322B 10DC 27830 DJNZ WRBOOT
27840 ;
27850 ; Verify this track
27860 ;
27870 IF @MOD4
322D 0610 27880 LD B,16 ;16 sector just written
322F 110000 27890 LD DE,0 ; on track 0
27900 ENDIF
27910 IF @MOD2
27920 LD A,(PROTSEC+1)
27930 LD B,5
27940 LD DE,(CKPROT2)
27950 OR A
27960 JR Z,NBTSEC1
27970 AND D
27980 JR Z,NBTSEC1
27990 LD B,16
28000 NBTSEC1 LD E,0
28010 ENDIF
3232 CD7728 28020 VRBOOT CALL VERSEC ;Verify a boot sector
3235 C29726 28030 JP NZ,EXIT3 ;Quit on an error

```

Backup By Class

```

3238 10F8      28040      DJNZ      VRBOOT
                28050 ;
                28060 ;      Mod II check if cyl 0 to be formatted on dest
                28070 ;
                28080      IF      @MOD2
                28090      LD      DE,(CKPROT2)      ;Get sysinfo sector
                28100      LD      A,(PROTSEC+1)
                28110      AND      D
                28120      JR      Z,COPY0E      ;Go if yes
                28130 OKWRT0 CALL      PMTSRC      ;Get source disk
                28140      CALL      READ0      ;Read cyl 0
                28150      JP      NZ,EXIT3      ;Go on disk error
                28160      CALL      PMTDST      ;Get dest disk
                28170      CALL      FORMAT0      ;Format cyl
                28180      JP      NZ,EXIT3      ;Go on disk error
                28190 ;
                28200 ;      Setup new track length into boot data
                28210 ;
                28220      LD      HL,(BUFFER$)      ;Get I/O buffer
                28230      PUSH     HL      ;Save start
                28240      INC      HL      ;+1
                28250      INC      HL      ;+2 (dir cyl)
                28260      LD      A,(IY+9)      ;Get dir cyl
                28270      LD      (HL),A      ;To buffer
                28280      INC      HL      ;+3 (boot step rate)
                28290      LD      A,(BSCLS+1)      ;Get step rate
                28300      AND      3      ;Step rate only
                28310      LD      (HL),A      ;Load into buffer
                28320      INC      HL      ;Bump
                28330      LD      A,(IY+7)      ;Get data
                28340      AND      1FH      ;Highest sector #
                28350      INC      A      ;Sectors / track
                28360      LD      (HL),A      ;To buffer
                28370      INC      HL      ;Bump
                28380      LD      A,(IY+3)      ;Get data
                28390      ADD      A,A      ;Density => bit 7
                28400      AND      80H      ;Keep only
                28410      LD      (HL),A      ;To buffer
                28420      POP      HL      ;HL => buffer start
                28430      LD      D,H      ;Pass to DE
                28440      LD      E,L      ;DE => buffer start
                28450      LD      BC,80H      ;Buffer length
                28460      ADD      HL,BC      ;HL => dest
                28470      EX      DE,HL      ;HL=>source, DE=>dest
                28480      LDIR      ;Copy sector 0 => sec 1
                28490      CALL      PMTDST      ;Re-fetch DCT
                28500      CALL      WRITE0      ;Write the cylinder
                28510      JP      NZ,EXIT3      ;Go on disk error
                28520 COPY0E EQU      $
                28530      ENDIF
                28540 ;
                28550 ;      Routine to perform the file copy to destination
                28560 ;
323A 11282F    28570 DOFIL0 LD      DE,OPENIT      ;Check the name
323D           28580      @RENAM
323D 3E38      00219      LD      A,56
323F EF        00220      RST      40
3240 0600      28590      LD      B,0      ;Lr1 = 256
3242 CD3D34    28600      CALL     GETSRC      ;Prompt source & set fcb

```

Backup By Class

```

3245 2A1626 28610 LD HL,(BUFFER$) ;Get buffer addr
3248 28620 @@FLAGS
3248 3E65 00221 LD A,101
324A EF 00222 RST 40
324B FDCB12C6 28630 SET 0,(IY+'S'-'A') ;Inhibit file open bit
324F 28640 @@OPEN ;Open the source file
324F 3E3B 00223 LD A,59
3251 EF 00224 RST 40
3252 C29726 28650 JP NZ,EXIT3 ;Quit on open error
28660 ;
28670 ; Check if source file can fit on destination disk
28680 ;
3255 2A2426 28690 LD HL,(FCB1$+12) ;P/u ERN
3258 110000 28700 SIZSAV LD DE,$-$ ;P/u disk capacity
325B AF 28710 XOR A
325C ED52 28720 SBC HL,DE ;If < size, then OK
325E 3809 28730 JR C,SIZOK
3260 21FC34 28740 LD HL,SIZBIG$ ; else file to big
3263 28750 @@LOGOT ;Inform user & continue
00225 IFEQ 00H,1
00226 LD HL,
00227 ENDIF
3263 3E0C 00228 LD A,12
3265 EF 00229 RST 40
3266 C32A2F 28760 JP SCNH1 ;Loop back for another file
3269 11282F 28770 SIZOK LD DE,OPENIT ;Check the name
326C 28780 @@RENAM
326C 3E38 00230 LD A,56
326E EF 00231 RST 40
326F 0600 28790 LD B,0 ;Lr1 = 256
3271 CD4634 28800 CALL GETDST ;Prompt dest & set fcb
3274 2A1626 28810 LD HL,(BUFFER$) ;Get buffer addr
3277 28820 @@INIT ;Init the dest
3277 3E3A 00232 LD A,58
3279 EF 00233 RST 40
327A 2807 28830 JR Z,LRL0K ;If no error, cont.
327C FE2A 28840 CP 42 ;Was it LRL error?
327E 2803 28850 JR Z,LRL0K ;Ignore if so
3280 C39726 28860 JP EXIT3 ; else real error, abort
3283 3A3F26 28870 LRL0K LD A,(FCB2$+7) ;P/u DEC of dest
3286 32FE32 28880 LD (DOFIL1+1),A
3289 ED4B2426 28890 LD BC,(FCB1$+12) ;P/u ERN & ck for enuf
328D CDF333 28900 CALL WRERN ; dest space on disk
3290 C1 28910 POP BC ;Recover DEC
3291 68 28920 LD L,B ;Reset HL to dir
3292 262C 28930 LD H,BUF2$<-8
3294 C5 28940 PUSH BC ;Save DEC
3295 2806 28950 JR Z,DOFIL02 ;Go if there was room
3297 CD1A27 28960 CALL PMTSRC ; else make source current, loop
329A C32A30 28970 JP DONAM ; back because dest was swapped
329D 7D 28980 DOFIL02 LD A,L ;Check if date current
329E E6E0 28990 AND 0E0H ;Index to proper direc
32A0 3C 29000 INC A
32A1 6F 29010 LD L,A
32A2 CB66 29020 BIT 4,(HL) ;Check if bit set
32A4 2803 29030 JR Z,$+5
32A6 323A2F 29040 LD (SETBIT),A
29050 ;
32A9 210000 29060 LD HL,0

```

Backup By Class

```

32AC 224426 29070 LD (FCB2$+12),HL ;Set dest ERN to 0
32AF 29080 @@REW ;Rewind the dest
32AF 3E44 00234 LD A,68
32B1 EF 00235 RST 40
32B2 2A1626 29090 DOFIL03 LD HL,(BUFFER$) ;Buffer addr
32B5 221B26 29100 DOFIL04 LD (FCB1$+3),HL ;Set buffer addr in fcb
32B8 CD3D34 29110 CALL GETSRC ;Prompt source & set fcb
32BB 29120 @@READ ;Read a source file sector
32BB 3E43 00236 LD A,67
32BD EF 00237 RST 40
32BE 280B 29130 JR Z,DOFIL05 ;Go if no error
32C0 FE1C 29140 CP 1CH ;Eof?
32C2 2824 29150 JR Z,DOFIL09 ;Yes, finished loading
32C4 FE1D 29160 CP 1DH ;Nrn > ern?
32C6 2820 29170 JR Z,DOFIL09 ;Also means load done
32C8 C39726 29180 JP EXIT3 ;Abort on any other error
32CB 24 29190 DOFIL05 INC H ;Bump the buffer ptr
32CC 7C 29200 LD A,H
32CD FE00 29210 DOFIL06 CP $-$ ;Test out of memory
32CF 20E4 29220 JR NZ,DOFIL04 ;Loop if more room
32D1 2A1626 29230 LD HL,(BUFFER$) ;P/u buffer start
32D4 223B26 29240 DOFIL07 LD (FCB2$+3),HL ; & set into dest fcb
32D7 CD4634 29250 CALL GETDST ;Prompt dest & set fcb
32DA 29260 @@VER ;Write dest w/verify
32DA 3E49 00238 LD A,73
32DC EF 00239 RST 40
32DD C29726 29270 JP NZ,EXIT3 ;Quit on error
32E0 24 29280 INC H ;Bump buffer page
32E1 7C 29290 LD A,H
32E2 FE00 29300 DOFIL08 CP $-$ ;Out of memory?
32E4 20EE 29310 JR NZ,DOFIL07 ;Write another if not
32E6 18CA 29320 JR DOFIL03 ; else back to loading
29330 ;
29340 ; Reached the end of the source file
29350 ;
32E8 CDD433 29360 DOFIL09 CALL LSTBUF ;Write remaining buffer
32EB 2A2026 29370 LD HL,(FCB1$+8) ;P/u DEC & LRL
32EE 224026 29380 LD (FCB2$+8),HL ; & stuff into dest
32F1 CD4634 29390 CALL GETDST ;Set for dest fcb
32F4 29400 @@CLOSE ;Close 'er up
32F4 3E3C 00240 LD A,60
32F6 EF 00241 RST 40
32F7 C29726 29410 JP NZ,EXIT3 ;Abort on close error
29420 ;
29430 ; Now remove the mod flag from destination
29440 ; and do CLONE function
29450 ;
32FA FD5609 29460 LD D,(IY+9) ;P/u dir cyl
32FD 0600 29470 DOFIL11 LD B,$-$ ;P/u DEC
32FF 78 29480 LD A,B ;Pt to dir sector
3300 E61F 29490 AND 1FH
3302 C602 29500 ADD A,2 ;Bypass GAT and HIT
3304 5F 29510 LD E,A
3305 D5 29520 PUSH DE ;Save cyl/sect
3306 2A1626 29530 LD HL,(BUFFER$) ;P/u buffer addr
3309 CD7228 29540 CALL RDSEC ;Read the dir sect
330C FE06 29550 CP 6 ;Proper errcod?
330E 3E11 29560 LD A,17 ;Init "Dir read error
3310 C29726 29570 JP NZ,EXIT3

```

Backup By Class

```

3313 78      29580      LD      A,B          ;Pt to dir record
3314 E6E0    29590      AND      0E0H
3316 5F      29600      LD      E,A          ;Pt to DIR lo order
3317 3A1726  29610      LD      A,(BUFFER$+1) ;P/u hi order buffer pos
331A 57      29620      LD      D,A
331B E1      29630      POP      HL
331C C1      29640      POP      BC          ;P/u DEC & buffer of src
331D C5      29650      PUSH     BC
331E E5      29660      PUSH     HL
331F 78      29670      LD      A,B          ;Get source DEC
3320 E6E0    29680      AND      0E0H          ; and pt to the direc
3322 6F      29690      LD      L,A          ; of the current file
3323 262C    29700      LD      H,BUF2$<-8
3325 2C      29710      INC      L          ;Pt to mod flag byte
3326 CBB6    29720      RES      6,(HL)        ;Reset the MOD bit
3328 2D      29730      DEC      L          ;Point to DIR+0
3329 010500  29740      LD      BC,5          ;Transfer up thru
332C EDB0    29750      LDIR                     ; DIR+4
332E 7B      29760      BYSPACE LD      A,E          ;Point DE to the dest
332F C60B    29770      ADD      A,11          ; password fields
3331 5F      29780      LD      E,A
3332 7D      29790      LD      A,L          ;Point HL to the source
3333 C60B    29800      ADD      A,11          ; password fields
3335 6F      29810      LD      L,A
3336 010400  29820      LD      BC,4          ;Move both pswds
3339 EDB0    29830      LDIR
333B 2A1626  29840      LD      HL,(BUFFER$)    ;P/u buffer addr
333E D1      29850      POP      DE          ;Rcvr cyl/sect
333F CD6D28  29860      CALL   WRSYS          ;Write back
3342 3E12    29870      LD      A,18          ;Init "Dir write error
3344 C29726  29880      JP      NZ,EXIT3        ;Quit on error
29890 ;
29900 ;      Attempt to clear mod flag of source
29910 ;
3347 3E00    29920      DOFIL12 LD      A,0          ;Test for write prot src
3349 B7      29930      OR       A          ;Which implies, can't
334A C22A2F  29940      JP      NZ,SCNH1        ; clear mod flags
334D C1      29950      POP      BC          ;P/u DEC of source
334E C5      29960      PUSH     BC
334F 78      29970      LD      A,B          ;Clear mod flag on source
3350 E6E0    29980      AND      0E0H          ;Dir sector is resident
3352 3C      29990      INC      A          ;In a buffer at BUF2
3353 6F      30000      LD      L,A
3354 262C    30010      LD      H,BUF2$<-8
3356 CBB6    30020      RES      6,(HL)        ;Reset mod bit
3358 CD1A27  30030      CALL   PMT SRC        ;Set for source i/o
335B FD5609  30040      LD      D,(IY+9)      ;P/u dir cyl
335E 78      30050      LD      A,B          ;Pt to dir sect of source
335F E61F    30060      AND      1FH
3361 C602    30070      ADD      A,2          ;Adjust for GAT and HIT
3363 5F      30080      LD      E,A
3364 21002C  30090      LD      HL,BUF2$
3367 CD6D28  30100      CALL   WRSYS          ;Write it back
336A CA2A2F  30110      JP      Z,SCNH1        ;Back on good write
336D FE0F    30120      CP      15          ;Accept only "write prot error
336F 3E12    30130      LD      A,18          ;Any other, "Dir write error
3371 C29726  30140      JP      NZ,EXIT3        ; and quit
3374 3EFF    30150      LD      A,0FFH        ;Turn off clear mod
3376 324833  30160      LD      (DOFIL12+1),A ; flag test

```


Backup By Class

```

3379          30170      @@LOGOT CCMOD$          ;"can't clear...
              00242      IFEQ      01H,1
3379 21C629    00243      LD        HL,CCMOD$
              00244      ENDIF
337C 3E0C      00245      LD        A,12
337E EF        00246      RST       40
337F C32A2F    30180      JP        SCNH1          ;Loop to next file
              30190      ;
              30200      ;      Routine to compare HL to DE, ret Z if equal
              30210      ;
3382 7C        30220 CPHLDE LD        A,H          ;Test H=D
3383 92        30230      SUB       D
3384 C0        30240      RET       NZ          ;Back if not
3385 7D        30250      LD        A,L          ;Test L=E
3386 93        30260      SUB       E
3387 C9        30270      RET              ;Back with condition
              30280      ;
              30290      ;      Routine to construct filespec from name/ext
              30300      ;
3388 3E3A      30310 MAKSPC LD        A,':'        ;Prepare for drivespec
338A 12        30320      LD        (DE),A
338B 13        30330      INC       DE
338C D5        30340      PUSH     DE          ;Save pointer
338D 3A7B27    30350      LD        A,(DSTD RV$+1) ;P/u dest drive #
3390 E607      30360      AND       7          ;Cvrt to ASCII
3392 C630      30370      ADD       A,'0'
3394 12        30380      LD        (DE),A      ; & stuff at filespec end
3395 13        30390      INC       DE
3396 3E03      30400      LD        A,3          ;Terminate with ETX
3398 12        30410      LD        (DE),A
3399 211826    30420      LD        HL,FCB1$     ;Copy source fcb to
339C 113826    30430      LD        DE,FCB2$     ; dest fcb
339F 012000    30440      LD        BC,32
33A2 EDB0      30450      LDIR
33A4 D1        30460      POP       DE          ;Rcvr where source spec
33A5 3A0E27    30470      LD        A,(SRC DRV$+1) ;P/u source drive #
33A8 E607      30480      AND       7          ;Cvrt to ASCII
33AA C630      30490      ADD       A,'0'
33AC 12        30500      LD        (DE),A      ;Stuff in dest fcb
33AD C9        30510      RET
              30520      ;
              30530      ;      Routine to extract date from directory
              30540      ;
33AE 7E        30550 UNPACK LD        A,(HL)     ;P/u DIR+1
33AF E60F      30560      AND       0FH         ;Remove flags
33B1 57        30570      LD        D,A         ;Save month
33B2 23        30580      INC       HL         ;Pt to DIR+2
33B3 7E        30590      LD        A,(HL)     ;P/u day and year
33B4 E6F8      30600      AND       0F8H        ;Strip year
33B6 5F        30610      LD        E,A         ;Save day in E
33B7 7E        30620      LD        A,(HL)     ;Get the year back
33B8 AB        30630      XOR       E          ;Strip the day
33B9 0F        30640      RRCA              ;Shift year to 5-7
33BA 0F        30650      RRCA
33BB 0F        30660      RRCA
33BC B2        30670      OR        D          ;Merge with month
33BD 57        30680      LD        D,A
33BE C9        30690      RET
              30700      ;

```

Backup By Class

```

30710 ; Write the GAT back to disk
30720 ;
33BF 2E00 30730 WRGAT LD L,0 ;HL to start of buffer
33C1 CD6D28 30740 CALL WRSYS ;Write dir sector
33C4 3E15 30750 LD A,21 ;Init GAT write error
33C6 C29726 30760 JP NZ,EXIT3 ; and quit on error
33C9 CD7728 30770 CALL VERSEC ;Verify good write
33CC FE06 30780 CP 6 ;Expect error 6
33CE 3E14 30790 LD A,20 ;Init GAT read error
33D0 C29726 30800 JP NZ,EXIT3 ;Quit on any other error
33D3 C9 30810 RET

30820 ;
30830 ; Write last buffer if needed
30840 ;
33D4 3A1726 30850 LSTBUF LD A,(BUFFER$+1) ;P/u hi order buffer start
33D7 BC 30860 CP H ;Are we there now?
33D8 C8 30870 RET Z ;Back if so, nothing loaded
33D9 3E00 30880 LSTBUF1 LD A,$-$ ;P/u last available page
33DB BC 30890 CP H ;There now?
33DC C8 30900 RET Z ;Already written if so
33DD 44 30910 LD B,H ;Need to write to this page
33DE 2A1626 30920 LD HL,(BUFFER$) ;P/u buffer start
33E1 223B26 30930 LSTBUF2 LD (FCB2$+3),HL ; and put in dest fcb
33E4 CD4634 30940 CALL GETDST ;Prompt dest
33E7 30950 @@VER ;Write with verify
33E7 3E49 00247 LD A,73
33E9 EF 00248 RST 40
33EA C29726 30960 JP NZ,EXIT3 ;Quit on bad write
33ED 24 30970 INC H ;Bump buffer page
33EE 7C 30980 LD A,H
33EF B8 30990 CP B ;At the end?
33F0 20EF 31000 JR NZ,LSTBUF2 ;Loop if more
33F2 C9 31010 RET

31020 ;
31030 ; Check if enough space on destination disk
31040 ;
33F3 78 31050 WRERN LD A,B ;If ERN = 0, don't
33F4 B1 31060 OR C ; write a ERN
33F5 C8 31070 RET Z
33F6 0B 31080 DEC BC ;Adjust for 0 offset
33F7 CD4634 31090 CALL GETDST ;Prompt dest
33FA D5 31100 PUSH DE ;Save fcb pointer
33FB 31110 @@POSN ;Position to end
33FB 3E42 00249 LD A,66
33FD EF 00250 RST 40
33FE 2A1626 31120 LD HL,(BUFFER$) ;P/u buffer addr
3401 54 31130 LD D,H ;Construct a format
3402 5D 31140 LD E,L ; sector of all X'E5's
3403 13 31150 INC DE
3404 01FF00 31160 LD BC,255
3407 36E5 31170 LD (HL),0E5H
3409 EDB0 31180 LDIR
340B D1 31190 POP DE ;Rcvr fcb ptr
340C 31200 @@VER ;Write with verify
340C 3E49 00251 LD A,73
340E EF 00252 RST 40
340F C8 31210 RET Z ;Ret if no error
3410 FE1B 31220 CP 27 ;Disk Full?
3412 2026 31230 JR NZ,NOTDF ;No - quit on real error

```

Backup By Class

```

3414          31240      @REMOV          ;Remove what can't fit
3414 3E39      00253      LD      A,57
3416 EF        00254      RST      40
3417 FDCB035E 31250      BIT      3,(IY+3)      ;Is this a rigid disk?
341B 280B      31260      JR      Z,NOTHARD      ;Go if not
341D FDCB0356 31270      BIT      2,(IY+3)      ;Shown as Removable?
3421 2805      31280      JR      Z,NOTHARD      ;Prompt disk swap if so
3423 217D34    31290      LD      HL,FULDRV$      ;Prepare disk full error
3426 183A      31300      JR      DOING1
3428          31310      NOTHARD @@FLAGS
3428 3E65      00255      LD      A,101
342A EF        00256      RST      40
342B FDCB126E 31320      BIT      5,(IY+'S'-'A') ;Can't switch while DOing
342F 202E      31330      JR      NZ,DOING
3431 218B34    31340      LD      HL,NEWDISK      ;"disk full, enter new...
3434 CDD327    31350      CALL    FLASH
3437 F601      31360      OR      1              ;Show switched dest
3439 C9        31370      RET
343A          31380      NOTDF EQU      $
343A C39726    31390      JP      EXIT3          ;Error exit
          31400      ;
343D C5        31410      GETSRC PUSH    BC
343E 111826    31420      LD      DE,FCB1$      ;Pt to source FCB
3441 CD1A27    31430      CALL    PMTSRC      ;Show source is current
3444 C1        31440      POP     BC          ; for disk I/O
3445 C9        31450      RET
          31460      ;
3446 C5        31470      GETDST PUSH    BC
3447 113826    31480      LD      DE,FCB2$      ;Pt to dest FCB
344A CD8727    31490      CALL    PMTDST      ;Show dest is current
344D C1        31500      POP     BC          ; for disk I/O
344E C9        31510      RET
          31520      ;
344F FD5609    31530      HITRD LD      D,(IY+9)      ;P/u dir cyl of source
3452 1E01      31540      LD      E,1          ;Read HIT
3454 210036    31550      LD      HL,HITBUF      ;Into HIT buffer
3457 CD7228    31560      CALL    RDSEC
345A FE06      31570      CP      6              ;Errcod correct?
345C 3E17      31580      LD      A,17H          ;Init "HIT read error
345E C9        31590      RET          ;Return w/condition
          31600      ;
345F 21CA34    31610      DOING LD      HL,DOMSG
3462 C3AF26    31620      DOING1 JP      EXIT4
          31630      ;
3465 1D        31640      CPYFIL$ DB      29,'Copying file: ',3
          43 6F 70 79 69 6E 67 20
          66 69 6C 65 3A 20 03
3475 42        31650      QUERY DB      'Backup ',3
          61 63 6B 75 70 20 03
347D 44        31660      FULDRV$ DB      'Disk is full ',CR
          69 73 6B 20 69 73 20 66
          75 6C 6C 20 0D
348B 44        31670      NEWDISK DB      'Disk is full - Insert new formatted '
          69 73 6B 20 69 73 20 66
          75 6C 6C 20 2D 20 49 6E
          73 65 72 74 20 6E 65 77
          20 66 6F 72 6D 61 74 74
          65 64 20
34AF 64        31680      DB      'destination disk, <ENTER>',29,3

```

Backup By Class

```

        65 73 74 69 6E 61 74 69
        6F 6E 20 64 69 73 6B 2C
        20 3C 45 4E 54 45 52 3E
        1D 03
34CA 44          31690 DOMSG    DB      'Disk is full! - Can''t switch '
        69 73 6B 20 69 73 20 66
        75 6C 6C 21 20 2D 20 43
        61 6E 27 74 20 73 77 69
        74 63 68 20
34E7 77          31700          DB      'while <D0> in effect',CR
        68 69 6C 65 20 3C 44 4F
        3E 20 69 6E 20 65 66 66
        65 63 74 0D
34FC 20          31710 SIZBIG$ DB      ' File is larger than destination '
        20 46 69 6C 65 20 69 73
        20 6C 61 72 67 65 72 20
        74 68 61 6E 20 64 65 73
        74 69 6E 61 74 69 6F 6E
        20
351E 63          31720          DB      'capacity - backup is bypassed',CR
        61 70 61 63 69 74 79 20
        2D 20 62 61 63 6B 75 70
        20 69 73 20 62 79 70 61
        73 73 65 64 0D
353C 43          31730 NOTSYS$ DB      'Can''t create SYSTEM disk - '
        61 6E 27 74 20 63 72 65
        61 74 65 20 53 59 53 54
        45 4D 20 64 69 73 6B 20
        2D 20
3557 64          31740          DB      'directory slots in use',CR
        69 72 65 63 74 6F 72 79
        20 73 6C 6F 74 73 20 69
        6E 20 75 73 65 0D
356E 7D          31750 QMARK$  DB      '} ? ',3
        20 3F 20 03
3573 4A          31760 MONTBL  DM      'JanFebMarAprMayJunJulAugSepOctNovDec '
        61 6E 46 65 62 4D 61 72
        41 70 72 4D 61 79 4A 75
        6E 4A 75 6C 41 75 67 53
        65 70 4F 63 74 4E 6F 76
        44 65 63
3597 A2          31770 SYSDEC  DB      0A2H,0C4H,2EH,2FH,2CH,2DH,2AH,2BH
        C4 2E 2F 2C 2D 2A 2B
359F 28          31780          DB      28H,29H,26H,27H,27H,0A7H,26H,0A6H
        29 26 27 27 A7 26 A6
        31790 ;
35A7 00          31800          DC      64,0          ;PATCH space
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        00 00 00 00 00 00 00 00
        31810 ;
3600          31820          ORG      $<-8+1<+8
0100          31830 HITBUF  DS      256
        31840 ;

```

Backup By Class

3700

31850

SUBTTL '<Backup Misc. routines>'

Backup Misc. routines

```

31870 ;
0900 31880 CLSSIZ EQU    $-BACKUP
31890 ;
31900 ;      Establish PC for rest of BACKUP initialization
31910 ;
4100 31920      ORG    CORE$+MIRSIZ+CLSSIZ
4100 31930      LORG    $      ;No offset here
31940 ;
31950 ;      Shift in Mirror or By-file module
31960 ;
4100 3E00 31970 CLSTST LD    A,0      ;Non-zero if any option
4102 B7 31980      OR    A
4103 C21341 31990      JP    NZ,MVBYCLS ;Bypass if special
4106 210032 32000      LD    HL,MIRBU ;Move in standard code
4109 11002E 32010      LD    DE,BACKUP
410C 010006 32020      LD    BC,MIRSIZ
410F EDB0 32030      LDIR
4111 1846 32040      JR    SETBFR
32050 ;
4113 3ACF27 32060 MVBYCLS LD    A,(SXORD+1) ;Restrict by class
4116 B7 32070      OR    A ; if a single drive
4117 2009 32080      JR    NZ,MVBYC1
4119 211B44 32090      LD    HL,CLS1DB$ ;Can't by class on 1 drv
411C 32100 MOVNOT @@DSPLY ;Display the error
00257      IFEQ    00H,1
00258      LD    HL,
00259      ENDIF
411C 3E0A 00260      LD    A,10
411E EF 00261      RST    40
411F C3AC26 32110      JP    ABRTBU ; and abort the backup
32120 ;
4122 3ACA26 32130 MVBYC1 LD    A,(XPARM$+1) ;By class backup requires
4125 B7 32140      OR    A ; either non (X) or residency
4126 2826 32150      JR    Z,MVBYC2 ; of SYS 2, 3, 10, and 12
4128 110000 32160 RESLOC LD    DE,$-$ ;Store location (RES$)
412B 7B 32170      LD    A,E
412C B2 32180      OR    D ;Check if there
412D 214744 32190      LD    HL,RESREQ$ ;Init "Must be resident
4130 28EA 32200      JR    Z,MOVNOT ;Error if not in use
4132 D5 32210      PUSH    DE ;OK, it's in use,
4133 DDE1 32220      POP    IX ; are all modules
4135 DD7E09 32230      LD    A,(IX+2*2+5) ; present and accounted
4138 B7 32240      OR    A ;SYS2 resident?
4139 28E1 32250      JR    Z,MOVNOT
413B DD7E0B 32260      LD    A,(IX+3*2+5) ;Is SYS3 resident?
413E B7 32270      OR    A
413F 28DB 32280      JR    Z,MOVNOT
4141 DD7E19 32290      LD    A,(IX+10*2+5) ;Is SYS10 resident?
4144 B7 32300      OR    A
4145 28D5 32310      JR    Z,MOVNOT
4147 DD7E1D 32320      LD    A,(IX+12*2+5) ;Is SYS12 resident?
414A B7 32330      OR    A
414B CA1C41 32340      JP    Z,MOVNOT
414E 210038 32350 MVBYC2 LD    HL,CLSBUS ;Move in special code
4151 11002E 32360      LD    DE,BACKUP
4154 010009 32370      LD    BC,CLSSIZ
4157 EDB0 32380      LDIR
4159 1B 32390 SETBFR DEC    DE ;Set the buffer
415A 14 32400      INC    D ; one page above the code
415B 1E00 32410      LD    E,0

```

Backup Misc. routines

```

415D ED531626 32420 LD (BUFFER$),DE ; and save starting posn
4161 C3002E 32430 JP BACKUP
          32440 ;
          32450 ; Routine to get password
          32460 ;
4164 CD6D41 32470 GETMPW CALL GMPW1
4167 3EE4 32480 LD A,0E4H ;Get SYS2 for hash
4169 EF 32490 RST 28H
          32500 ;
416A 3E84 32510 GETSYS2 LD A,84H ;Load SYS2, no function
416C EF 32520 RST 28H
          32530 ;
416D 7A 32540 GMPW1 LD A,D ;Pswd entered as parm?
416E B3 32550 OR E
416F 281A 32560 JR Z,GMPW3 ;Prompt if not
4171 21002D 32570 LD HL,BUF3$
4174 E5 32580 PUSH HL
4175 0608 32590 LD B,8
4177 1A 32600 GMPW2 LD A,(DE) ;P/u pswd character
4178 FE0D 32610 CP CR ;At end of line?
417A 282A 32620 JR Z,GMPW4 ;Space out if yes
417C FE2C 32630 CP ',' ;Comma separator?
417E 2826 32640 JR Z,GMPW4
4180 FE22 32650 CP '"' ;Closing quote?
4182 2822 32660 JR Z,GMPW4
4184 13 32670 INC DE
4185 77 32680 LD (HL),A ;Xfer the character
4186 23 32690 INC HL
4187 10EE 32700 DJNZ GMPW2
4189 1820 32710 JR GMPW5
          32720 ;
          32730 ; Not entered as parm, grab from keyboard
          32740 ;
418B 32750 GMPW3 @@DSPLY ;Display request
          00262 IFEQ 00H,1
          00263 LD HL,
          00264 ENDIF
418B 3E0A 00265 LD A,10
418D EF 00266 RST 40
418E 010008 32760 LD BC,8<8 ;Max 8 chars input
4191 21002D 32770 LD HL,BUF3$ ;Point to buffer
4194 E5 32780 PUSH HL
4195 32790 @@KEYIN ;Grab password
4195 3E09 00267 LD A,9
4197 EF 00268 RST 40
4198 DAAC26 32800 JP C,ABRTBU ;Abort on BREAK
419B EB 32810 EX DE,HL ;Buf start to DE
419C 2600 32820 LD H,0 ;Buf length to HL
419E 68 32830 LD L,B
419F 19 32840 ADD HL,DE ;Pt to 1st unused pos
41A0 3E08 32850 LD A,8 ;Calculate spaces needed
41A2 90 32860 SUB B
41A3 2806 32870 JR Z,GMPW5 ;Don't put any if 8 input
41A5 47 32880 LD B,A ;Set space counter
41A6 3620 32890 GMPW4 LD (HL),' '
41A8 23 32900 INC HL
41A9 10FB 32910 DJNZ GMPW4
41AB E1 32920 GMPW5 POP HL ;Rcvr pointer to buf
41AC E5 32930 PUSH HL

```

Backup Misc. routines

```

41AD 0608      32940      LD      B,8          ;Loop thru field
41AF 7E        32950 GMPW6 LD      A,(HL)
41B0 FE61      32960      CP      'a'
41B2 3806      32970      JR      C,GMPW7
41B4 FE7B      32980      CP      'z'+1
41B6 3002      32990      JR      NC,GMPW7
41B8 CBAE      33000      RES     5,(HL)        ;Lc -> UC
41BA 23        33010 GMPW7 INC     HL
41BB 10F2      33020      DJNZ    GMPW6
41BD D1        33030      POP     DE          ;Rcvr pointer to start
41BE C9        33040      RET
                33050 ;
                33060 ;      Check a drive for availability
                33070 ;
                33080 CKDRV
41BF 3AC827     33090      LD      A,(CURDSK+1) ;P/u drive spec
41C2 4F        33100      LD      C,A          ;Place in C
41C3 FD7E00     33110      LD      A,(IY+0)      ;P/u drive vector
41C6 FEC3      33120      CP      0C3H         ;Ck for enabled
41C8 C24242     33130      JP      NZ,CKDR5      ;Bypass if disabled
41CB E5        33140      PUSH    HL
41CC D5        33150      PUSH    DE
41CD FD7E06     33160      LD      A,(IY+6)      ;Make sure the current
41D0 FDBE05     33170      CP      (IY+5)        ; cylinder count is in range
41D3 D2DC41     33180      JP      NC,CKDRV1     ;Go if in range
41D6 CD5E28     33190      CALL    RESTOR        ;Restore drive
41D9 C24F42     33200      JP      NZ,CKDR7A     ;Go if error
                33210 ;
41DC FD5605     33220 CKDRV1 LD      D,(IY+5)    ;P/u current track
41DF 1E00       33230      LD      E,0          ;Set for sector 0
41E1           33240      @@SEEK                ;Set track info to FDC
41E1 3E2E      00269      LD      A,46
41E3 EF        00270      RST      40
41E4 2069      33250      JR      NZ,CKDR7A     ;Go if error
41E6 CD6328     33260      CALL    RSELECT      ;Wait until not busy
41E9 2064      33270      JR      NZ,CKDR7A     ;Not there - ret NZ
41EB FDCB035E   33280      BIT     3,(IY+3)      ;If hard drive, bypass
41EF 2047      33290      JR      NZ,CKDR3A     ; GAT data update
41F1 FDCB0466   33300      BIT     4,(IY+4)      ;If "ALIEN" by pass
41F5 201E      33310      JR      NZ,CKDR2B     ; test of index pulses
                33320      IF      @MOD4
41F7 3E09      33330      LD      A,09          ;Set MSB of count down
41F9 F3        33340      DI
                33350      ENDIF
                33360      IF      @MOD2
                33370      LD      A,20
                33380      ENDIF
41FA 320642     33390 INTRON LD      (CDCNT+1),A ;Store in 'LD H' instruction
41FD 212000     33400      LD      HL,0020H      ;Set up count (short)
                33410 ;
                33420 ;      Test for diskette in drive & rotating
                33430 ;
4200 CD4342     33440 CKDR1  CALL    INDEX      ;Test index pulse
4203 20FB      33450      JR      NZ,CKDR1      ;Jump on index
4205 2600      33460 CDCNT  LD      H,00H       ;CKDRV counter (long)
                33470      ;Count set from above
4207 CD4342     33480 CKDR2  CALL    INDEX      ;Test index pulse
420A 28FB      33490      JR      Z,CKDR2      ;Jump on no index
                33500      IF      @MOD4

```


Backup Misc. routines

```

420C FB      33510      EI                      ;OK for INTs now
              33520      ENDIF
420D 212000  33530      LD      HL,0020H        ;Index off wait (short)
4210 CD4342  33540 CKDR2A CALL      INDEX
4213 20FB    33550      JR      NZ,CKDR2A        ;Jump on index
              33560      ;
              33570      ;      Diskette is rotating
              33580      ;
4215 F5      33590 CKDR2B PUSH      AF          ;Save FDC status
4216 FD5609  33600      LD      D,(IY+9)
4219 210046  33610      LD      HL,CKDRBUF      ;Point to HIT buffer
421C 5D      33620      LD      E,L            ;Sector 0 for GAT
421D         33630      @@RDSSC                ;Read the GAT
421D 3E55    00271      LD      A,85
421F EF      00272      RST      40
4220 202C    33640      JR      NZ,CKDR7        ;Jump on error
4222 2ACC46  33650      LD      HL,(CKDRBUF+0CCH) ;P/u excess tracks
4225 3E22    33660      LD      A,22H          ;Add offset
4227 85      33670      ADD      A,L
4228 FD7706  33680      LD      (IY+6),A        ;Max track # to DCT
422B FDCB04AE 33690      RES      5,(IY+4)      ;Set to side 0
422F CB6C    33700      BIT      5,H            ;Test double sided
4231 2804    33710      JR      Z,CKDR3        ;Jump if only single
4233 FDCB04EE 33720      SET      5,(IY+4)      ;Set for side 2
4237 F1      33730 CKDR3  POP      AF          ;Recover FDC status
4238 07      33740 CKDR3A RLCA                ;Shift write prot to 7
4239 FDB603  33750      OR      (IY+3)         ;Merge Soft WP bit
423C E680    33760      AND      80H           ;Strip all but 7
423E 87      33770      ADD      A,A           ;Write prot to carry flg
              33780      ;
423F         33790 CKDR4  EQU      $
423F FB      33800      EI
4240 D1      33810      POP      DE
4241 E1      33820      POP      HL
4242 C9      33830 CKDR5  RET
4243 7C      33840 INDEX  LD      A,H          ;Count down tries
4244 B5      33850      OR      L
4245 2807    33860      JR      Z,CKDR7        ;Error if counted out
4247 2B      33870      DEC      HL            ;Dec the count
4248 CD6328  33880      CALL      RSELECT      ;Check for index pulse
424B CB4F    33890      BIT      1,A           ;Test index
424D C9      33900      RET                    ;Back with condition
424E F1      33910 CKDR7  POP      AF
424F 3E08    33920 CKDR7A LD      A,8          ;Set Device not avail
4251 B7      33930      OR      A             ;Set NZ ret
4252 18EB    33940      JR      CKDR4          ;Leave
              33950      ;
              33960      ;      Data area
              33970      ;
              33980 PRMTBL$
0080         33990 VAL    EQU      80H
0040         34000 SW     EQU      40H
0020         34010 STR    EQU      20H
0010         34020 SGL    EQU      10H
4254 D3      34030      DB      'S'!80H
4255 63      34040      DB      SW!STR!3,'MPW',0
              4D 50 57 00
0005         34050 MPWRSP EQU      $-PRMTBL$-1
425A DA30    34060      DW      MPWPRM

```

Backup Misc. routines

```

425C 73      34070      DB      SW!STR!SGL!3,'SYS',0
      53 59 53 00
000C      34080 SYSRSP EQU      $-PRMTBL$-1
4261 102F    34090      DW      SYSPRM+1
4263 53      34100      DB      SW!SGL!3,'INV',0
      49 4E 56 00
0013      34110 INVRSP EQU      $-PRMTBL$-1
4268 192F    34120      DW      INVPRM+1
426A 53      34130      DB      SW!SGL!3,'MOD',0
      4D 4F 44 00
001A      34140 MODRSP EQU      $-PRMTBL$-1
426F 1226    34150      DW      MODPRM$
4271 55      34160      DB      SW!SGL!5,'QUERY',0
      51 55 45 52 59 00
0023      34170 QRSP   EQU      $-PRMTBL$-1
4278 1426    34180      DW      QPARM$
427A 41      34190      DB      SW!1,'X',0
      58 00
0028      34200 XRSP   EQU      $-PRMTBL$-1
427D CA26    34210      DW      XPARM$+1
427F 34      34220      DB      STR!SGL!4,'DATE',0
      44 41 54 45 00
0030      34230 DATRSP EQU      $-PRMTBL$-1
4285 D82E    34240      DW      DATPRM+1
4287 53      34250      DB      SW!SGL!3,'NEW',0
      4E 45 57 00
0037      34260 NEWRSP EQU      $-PRMTBL$-1
428C 0E26    34270      DW      NEWPRM$
428E 53      34280      DB      SW!SGL!3,'OLD',0
      4F 4C 44 00
003E      34290 OLDRSP EQU      $-PRMTBL$-1
4293 1026    34300      DW      OLDPRM$
4295 00      34310      NOP
      34320 ;
4296 53      34330 NOINDO$ DB      'Single drive backup invalid during'
      69 6E 67 6C 65 20 64 72
      69 76 65 20 62 61 63 6B
      75 70 20 69 6E 76 61 6C
      69 64 20 64 75 72 69 6E
      67
42B8 20      34340      DB      ' <D0> processing',CR
      3C 44 4F 3E 20 70 72 6F
      63 65 73 73 69 6E 67 0D
42C9 44      34350 NOFMT$ DB      'Destination disk not formatted'
      65 73 74 69 6E 61 74 69
      6F 6E 20 64 69 73 6B 20
      6E 6F 74 20 66 6F 72 6D
      61 74 74 65 64
42E7 20      34360      DB      ' - Backup aborted',CR
      2D 20 42 61 63 6B 75 70
      20 61 62 6F 72 74 65 64
      0D
42F9 42      34370 HELLO$ DB      'BACKUP'
      41 43 4B 55 50
42FF      34380 *GET   CLIENT:3
      34390 ;CLIENTS/ASM - File to establish sign-on headers
      34400 ;
42FF 20      34410      DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
      2D 20 36 2E 32 2E 30 20
      2D 20 43 6F 70 79 72 69

```

Backup Misc. routines

```

        67 68 74 20 31 39 38 32
        2F 38 33 2F 38 34 20 62
        79 20 4C 6F 67 69 63 61
        6C
4329 20      34420      DB      ' Systems, Inc.      ',10
        53 79 73 74 65 6D 73 2C
        20 49 6E 63 2E 20 20 20
        20 20 20 0A
            34430 ;
433E 41      34440      DB      'All Rights Reserved. Licensed 1982/83/84'
        6C 6C 20 52 69 67 68 74
        73 20 52 65 73 65 72 76
        65 64 2E 20 4C 69 63 65
        6E 73 65 64 20 31 39 38
        32 2F 38 33 2F 38 34
4366 20      34450      DB      ' to xxxxxxxxxxxxxxxxxxxx',10,13
        74 6F 20 78 78 78 78 78
        78 78 78 78 78 78 78 78
        78 78 78 78 78 0A 0D
437E 43      34460 LDOS$      DB      'Command executes only from DOS Ready',CR
        6F 6D 6D 61 6E 64 20 65
        78 65 63 75 74 65 73 20
        6F 6E 6C 79 20 66 72 6F
        6D 20 44 4F 53 20 52 65
        61 64 79 0D
43A3 50      34470 PRMERR$ DB      'Parameter error',CR
        61 72 61 6D 65 74 65 72
        20 65 72 72 6F 72 0D
43B3 53      34480 SRCNUM$ DB      'Source drive number ?      ',3
        6F 75 72 63 65 20 64 72
        69 76 65 20 6E 75 6D 62
        65 72 20 3F 20 20 20 20
        20 20 20 20 03
43D1 44      34490 DSTNUM$ DB      'Destination drive number ?      ',3
        65 73 74 69 6E 61 74 69
        6F 6E 20 64 72 69 76 65
        20 6E 75 6D 62 65 72 20
        3F 20 20 20 03
43EF 4E      34500 NODAT$  DB      'No date established',CR
        6F 20 64 61 74 65 20 65
        73 74 61 62 6C 69 73 68
        65 64 0D
4403 42      34510 CLASS$  DB      'Backup by class invoked',CR
        61 63 6B 75 70 20 62 79
        20 63 6C 61 73 73 20 69
        6E 76 6F 6B 65 64 0D
441B 0A      34520 CLS1DB$ DB      LF,'Single drive BACKUP invalid by files',CR
        53 69 6E 67 6C 65 20 64
        72 69 76 65 20 42 41 43
        4B 55 50 20 69 6E 76 61
        6C 69 64 20 62 79 20 66
        69 6C 65 73 0D
            34530      IF
4441 53      34540 RES$      DB      '.NOT.SMALL
        59 53 52 45 53      'SYSRES'      ;Terminate with LF
4447 0A      34550 RESREQ$ DB      LF,'This backup requires residency '
        54 68 69 73 20 62 61 63
        6B 75 70 20 72 65 71 75
        69 72 65 73 20 72 65 73

```

Backup Misc. routines

```

        69 64 65 6E 63 79 20
4467 6F      34560 DB      'of SYS's: 2, 3, 10 & 12.',CR
        66 20 53 59 53 27 73 3A
        20 32 2C 20 33 2C 20 31
        30 20 26 20 31 32 2E 0D
            34570      ENDIF
            34580      IF      SMALL
            34590 RESREQ$ DB      'Backup by class requires the us'
            34600      DB      'e of a SYSTEM diskette!      ',CR
            34610      ENDIF
4480 42      34620 RECON$ DB      'Backup-reconstruct invoked',CR
        61 63 6B 75 70 2D 72 65
        63 6F 6E 73 74 72 75 63
        74 20 69 6E 76 6F 6B 65
        64 0D
449B 43      34630 MIRROR$ DB      'Cylinder count differs - '
        79 6C 69 6E 64 65 72 20
        63 6F 75 6E 74 20 64 69
        66 66 65 72 73 20 2D 20
44B4 41      34640 DB      'Attempt mirror-image backup ? ',3
        74 74 65 6D 70 74 20 6D
        69 72 72 6F 72 2D 69 6D
        61 67 65 20 62 61 63 6B
        75 70 20 3F 20 03
44D3 4D      34650 PMTMPW$ DB      'Master password ?      ',3
        61 73 74 65 72 20 70 61
        73 73 77 6F 72 64 20 3F
        20 20 20 20 20 20 03
44EB 1F      34660 MAXDAYS DB      31,28,31,30,31,30,31,31,30,31,30,31
        1C 1F 1E 1F 1E 1F 1F 1E
        1F 1E 1F
44F7 42      34670 BADFMT$ DB      'Bad date format',CR
        61 64 20 64 61 74 65 20
        66 6F 72 6D 61 74 0D
4600      34680 CKDRBUF EQU      $<-8+1<8
0100      34690      DS      256
4607      34700 LAST      EQU      $
            00140 ;
4607      00150      SUBTTL <>
2E00      00160      END      BACKUP

```

\$A1	2E9E \$A2	2F1D \$EX4	2EBB
@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ABRTBU	26AC ABRTBU\$	2A0D ATTRIB	316C
AUTO	00E0 BACKUP	2E00 BACKUPA	2E09
BADFMT	314E BADFMT\$	44F7 BADMPW\$	28E6
BCK1	2E2B BCK2	2E3F BCK3	2E50
BCK4	2E6B BCK5	2E77 BCK6	2E9C
BOOTST\$	2600 BREAK	26AC BSCLS	3218
BSMIR	3005 BUCA0\$	29FC BUCORE\$	2A69
BUF1\$	2B00 BUF2\$	2C00 BUF3\$	2D00
BUFFER\$	2616 BYCLAS	30AB BYPASS	30B5
BYSFACE	332E CANTBU	2FEC CANTBU\$	2A1F
CCMOD\$	29C6 CDCNT	4205 CKBOOT	2FF7
CKCLA1	2F21 CKCLAS	2F0D CKDR1	4200
CKDR2	4207 CKDR2A	4210 CKDR2B	4215
CKDR3	4237 CKDR3A	4238 CKDR4	423F
CKDR5	4242 CKDR7	424E CKDR7A	424F
CKDRBUF	4600 CKDRV	41BF CKDRV1	41DC
CKDST	305F CKGAT	2FF2 CKINV	2FB7
CKNAM	2FC0 CKNAM0	2FCE CKNAM1	2FD8
CKNAM2	2FE5 CKOLD	30A2 CKSWDD	2889
CKTO	2EF8 CLASS\$	4403 CLSIDB\$	441B
CLSB	3800 CLSBU0	2EA9 CLSBU01	2ECC
CLSB1	2EDE CLSBU2	2EE4 CLSBU3	2EED
CLSB4	2EF6 CLSBU5	2F1A CLSFLG\$	2684
CLSSIZ	0900 CLSTST	4100 CNTBAK1	312C
CORE\$	3200 CPHLDE	3382 CPRID	2E36
CPRLOK	2EA6 CPYFIL\$	3465 CPYMSG	3147
CR	000D CURDSK	27C7 CVD1	31A8
CVD2	31AA CVD3	31B0 CVD7	31B9
CVTDEC	3196 CYL\$	3201 DAT	00D8
DATFLD\$	2678 DATPRM	2ED7 DATRSP	0030
DIFDST\$	298F DIFID\$	3233 DIFSRC	276B
DIFSRC\$	295D DI01	287A DIRERR	2692
DOBU	3161 DOFIL0	323A DOFIL02	329D
DOFIL03	32B2 DOFIL04	32B5 DOFIL05	32CB
DOFIL06	32CD DOFIL07	32D4 DOFIL08	32E2
DOFIL09	32E8 DOFIL11	32FD DOFIL12	3347
DOING	345F DOING1	3462 DOMSG	34CA
DONAM	302A DONAM1	3037 DONAM2	3041
DONAM4	304F DONAM5	3059 DOSYS1	31AB
DOSYS2	31FE DSTDFT	301A.DSTDIR	300E
DSTDRV\$	277A DSTNUM\$	43D1 DSTWP\$	28C1
DUCYL	2FA4 DUCYL\$	31D7 DUCYL1	2FA6
DUCYL2	2FE1 DUCYL2A	300B DUCYL2B	300D
DUCYL3	301A DUCYL4	3022 DUCYL5	3025
DUCYL6	3030 ERREXIT	26BA EX1	28B4
EX2	28BA EXIT	26BD EXIT1	2685
EXIT2	2695 EXIT3	2697 EXIT4	26AF
EXIT5	26C9 EXIT5A	26D8 EXIT5B	26DF
FCB1\$	2618 FCB2\$	2638 FCB3\$	2658
FCNT1	1111 FCNT2	1555 FLASH	27D3
FLASH0	27E3 FLS1	27F9 FLS2	2814
FLS4	2827 FLS5	2828 FLSH6	283D
FMPAKD\$	2680 FMT	0000 FRCDAT	2F01
FRCPMT	27C1 FTFLG\$	2601 FULDRV\$	347D
GETDAT	2F5D GETDAT1	2F7B GETDST	3446

GETGM	2F6D GETMPW	4164 GETSRC	343D
GETSYS2	416A GMPW1	416D GMPW2	4177
GMPW3	418B GMPW4	41A6 GMPW5	41AB
GMPW6	41AF GMPW7	41BA GODOIT	30A9
GOTDST	3047 GOTSRC	2FAA HELLO\$	42F9
HITBUF	3600 HITRD	344F IDMATCH	2E9A
INDEX	4243 INTRON	41FA INVPRM	2F18
INVRSP	0013 LAST	4607 LDCYL\$	31C2
LDCYL2	2F70 LDCYL3	2F7A LDCYL4	2F7D
LDCYL5	2F81 LDCYL6	2F88 LDCYL7	2F8D
LDCYL8	2F9B LDO\$S	437E LDTKS	2F30
LDTKS1	2F37 LF	000A LILBUF\$	2658
LOCK	0060 LRLOK	3283 LSTBUF	33D4
LSTBUF1	33D9 LSTBUF2	33E1 MAKSPC	3388
MATCHES	3029 MAXDAYS	44EB MFLG\$	260D
MIRBU	3200 MIRROR	30B7 MIRROR\$	449B
MIRSIZ	0600 MODPRM\$	2612 MODRSP	001A
MONTBL	3573 MOVID	308F MOVNOT	411C
MPWPRM	30DA MPWRSP	0005 MVBYC1	4122
MVBYC2	414E MVBYCLS	4113 NDSYS\$	26FB
NEWDISK	348B NEWPRM\$	260E NEWRSP	0037
NODAT\$	43EF NODOIT	2FBD NOFMT\$	42C9
NOINDO\$	4296 NOPRMPT	3149 NOTDF	343A
NOTHARD	3428 NOTMIR	31BC NOTMIR\$	3205
NOTSYS	2EE7 NOTSYS\$	353C OLDMPW	2E66
OLDMPW\$	3272 OLDPRM\$	2610 OLDRSP	003E
OPENIT	2F28 PACKID\$	3255 PACKNDO	2E97
PAKDAT	310B PARSDAT	3154 PASSWORD	42E0
PMTDD	303D PMTDST	2787 PMTDST\$	293A
PMTDST1	27B0 PMTMPW\$	44D3 PMTSRC	271A
PMTSRC\$	291C PMTSYS\$	28FE PMTYN	2E7E
PMTYN\$	32A4 PRMERR\$	43A3 PRMTBL\$	4254
PROT\$	2A4A PROTSEC	2897 PRS4	3180
PRSD1	3159 PRSD2	3169 PRSD3	317E
PRSPEC	30F0 PS1	30FE PSRC1	275F
PSRC3	2774 PSWD	00CE QM1	30C0
QMARK\$	356E QPARM\$	2614 QRSP	0023
QUERY	3475 RDBOOT	31EB RDSEC	2872
RECON	30AE RECON\$	4480 RES\$	4441
RESKFLG	284C RESLOC	4128 RESMF	30D0
RESMF1	30DC RESMF2	30F0 RESMF2A	314E
RESMF2B	3157 RESMF3	3162 RESMF4	316E
RESMF5	318D RESMF6	318F RESREQ\$	4447
RESTOR	285E RETCOD	26C1 RSELECT	2863
SCDAT1	30D2 SCDAT2	30E7 SCDAT4	311D
SCNH1	2F2A SCNH2	2F2E SCNH3	2F5C
SCNH4	2F9C SCNH4A	2FA9 SCNH5	2FF9
SCNH6	2FFA SCNH7	3015 SCNHIT	2F29
SELECT	2859 SET0	3008 SETBFR	4159
SETBIT	2F3A SETSYS	2EA2 SGL	0010
SHOPROT	308F SIZBIG\$	34FC SIZOK	3269
SIZSAV	3258 SMALL	0000 SPCFLD\$	2602
SPSAV	26BD SRCDFI	2F95 SRCDRV\$	270D
SRCNUM\$	43B3 STR	0020 STRDIR\$	3103
SVCTR	28A1 SW	0040 SXORD	27CE
SYSDEC	3597 SYSDRV\$	2700 SYSPRM	2F0F
SYSRSP	000C TKCAP	00CC TOEXIT1	2F59
TOPAKD\$	2682 TST5 8	304C TSTCAP	3097
TSTDIV	2855 TSTMFLG	2FF2 TSTMPW	30CF

UNPACK	33AE VAL	0080 VECYL\$	31EC
VECYL1	303C VECYL2	306E VECYL3	3078
VECYL4	307A VECYL5	307E VECYL6	3084
VERSEC	2877 VRBOOT	3232 WRBOOT	3209
WRBOOT1	321B WRBOOT2	3223 WRERN	33F3
WRGAT	33BF WRSEC	2868 WRSYS	286D
XPARAM\$	26C9 XRSP	0028 @@ABORT	6C65
@@ADTSK	6CF8 @@BANK	7210 @@BKSP	6EF0
@@BREAK	7226 @@CHNIO	6C50 @@CKBRKC	7274
@@CKDRV	6D4C @@CKEOF	6F05 @@CKTSK	6CE3
@@CLOSE	6EDB @@CLS	725E @@CMNDI	6C8F
@@CMNDR	6CA4 @@CTL	6AB4 @@DATE	6C26
@@DCSTAT	6D8B @@DEBUG	6CCE @@DECHEX	7190
@@DIRRD	70FD @@DIRWR	7112 @@DIV16	717B
@@DIV8	7166 @@DODIR	6D61 @@DSP	6A78
@@DSPLY	6B18 @@ERROR	6CB9 @@EXIT	6C7A
@@FEXT	706A @@FLAGS	71FA @@FNAME	707F
@@FSPEC	7055 @@GATRD	70E8 @@GATWR	7127
@@GET	6A8C @@GTDCB	70A9 @@GTDCT	7094
@@GTMOD	70BE @@HDFMT	6E33 @@HEX16	71CF
@@HEX8	71BA @@HEXDEC	71A5 @@HIGH\$	71E4
@@INIT	6EB1 @@KBD	6AF0 @@KEY	6A64
@@KEYIN	6B04 @@KLTSK	6D37 @@LOAD	702B
@@LOC	6F1A @@LOF	6F2F @@LOGGER	6B4F
@@LOGOT	6B64 @@MSG	6B9B @@MUL16	7151
@@MUL8	713C @@OPEN	6EC6 @@PARAM	6C11
@@PAUSE	6BFC @@PEOF	6F44 @@POSN	6F59
@@PRINT	6BB0 @@PRT	6AC8 @@PUT	6AA0
@@RAMDIR	6D76 @@RDSEC	6E09 @@RDSSC	70D3
@@READ	6F6E @@REMOV	6E9C @@RENAM	6E87
@@REW	6F83 @@RMTSK	6D0D @@RPTSK	6D22
@@RREAD	6F98 @@RSLCT	6DF4 @@RSTOR	6DB5
@@RUN	7040 @@RWRTT	6FAD @@SEEK	6DDF
@@SEEKSC	6FC2 @@SKIP	6FD7 @@SLCT	6DA0
@@STEPI	6DCA @@TIME	6C3B @@VDCTL	6BE7
@@VER	6FEC @@VRSEC	6E1E @@WEOF	7001
@@WHERE	6ADC @@WRITE	7016 @@WRSEC	6E48
@@WRSSC	6E5D @@WRTRK	6E72	

2E00 is the transfer address
00000 Total errors

NOTES:

NOTES:

CLICK/FLT - Sound click device filter

The Click filter can be used to generate a short clicking sound on the occurrence of all characters sent to a device, or on a specific character only. Click will always install itself in high memory, and will not attempt to load in the low driver zone. It is installed with the SET and FILTER Library commands.

```

00100 ;CLICK/ASM - Device Click Filter
0000 00110 TITLE <CLICK/FLT - LS-DOS 6.2>
00120 ;
00130 ;
00140 IF @MOD4
0048 00150 TONE EQU 48H
0018 00160 LEN EQU 18H
0090 00170 SNDPORT EQU 90H
00180 ENDF
00190 IF @MOD2
00200 LEN EQU 180H ;Length
00210 SNDPORT EQU 0A0H
00220 ENDF
00230 ;
0000 00240 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
0000 00250 *GET VALUES:3 ;Misc. equates
00320 ;VALUES/ASM - Version 6
00330 *LIST OFF
00420 *LIST ON
0000 00260 *GET COPYCOM:3 ;Copyright messages
004210 ; COPYCOM - File for Copyright COMMENT block
004220 ;
0000 004230 COM '<*(C) 1982,83,84 by LSI*>'
00270 ;
2400 00280 ORG 2400H
00290 ;
00300 START
2400 00310 @@CKBRKC
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00320 JR Z,STARTA ;Continue if no BREAK
2405 21FFFF 00330 LD HL,-1 ; set up abort RET
2408 C9 00340 RET
00350 ;
2409 ED730B25 00360 STARTA LD (EXIT+1),SP ;Save stack for error exit
240D CD1724 00370 CALL DOINIT ;Do initialization
2410 CD9624 00380 CALL INSTFLT ;Relocate/install filter
2413 210000 00390 NORMEX LD HL,0 ;Good exit
2416 C9 00400 RET
00410 ;
00420 ; Xfer DCB ptr to IX & stuff addrs' in driver
00430 ;
00440 ;
2417 D5 00440 DOINIT PUSH DE ;DE => DCB+0
2418 DDE1 00450 POP IX ;Xfer to IX
241A ED535F24 00460 LD (DCB),DE ;Xfer into header
00470 ;
00480 ; Sign-on
00490 ;
241E E5 00500 PUSH HL
241F 215125 00510 LD HL,HELLO$ ;Sign on message
2422 CDEC24 00520 CALL DSPLY
00530 ;
00540 ; Check PARMS and if entry from SET command
00550 ;
2425 111125 00560 LD DE,PRMTBL ;Point to parms
2428 E1 00570 POP HL ;Recover cmdline posn
2429 00580 @@PARAM ;Parse the parms
2429 3E11 00003 LD A,17

```

The Source	UTILITY Files	CLICK/FLT - LS-DOS 6.2	Page 00002
242B EF	00004	RST 40	
242C C2F224	00590	JP NZ,IOERR	;Exit on parm error
	00600 ;		
242F	00610	@@FLAGS	;IY => System Flags Base
242F 3E65	00005	LD A,101	
2431 EF	00006	RST 40	
2432 FDCB025E	00620	BIT 3,(IY+'C'-'A')	;System request?
2436 CAFD24	00630	JP Z,VIASET	;Install with SET
	00640 ;		
	00650 ;		
	00660 ;		Before anything - Make sure hi-mem is avail
2439 FDCB0246	00670	BIT 0,(IY+CFLAG\$)	;High memory available ?
243D C20125	00680	JP NZ,CANT	;No - display error
	00690 ;		
	00700 ;		Set up filter for CHAR if entered
	00710 ;		
2440 110000	00720	CHARPRM LD DE,00	;Char parm lands here
2443 7A	00730	LD A,D	;Check if entered and
2444 BB	00740	CP E	; is normal character
2445 C8	00750	RET Z	;Done if not entered
2446 FE00	00760	CP 0	;Check is MSB is altered
2448 3E2C	00770	LD A,44	;Init "Parameter error
244A C2F224	00780	JP NZ,IOERR	;Bad if so
	00790 ;		
244D 53	00800	LD D,E	;Set up CP nn
244E 1EFE	00810	LD E,0FEH	;Reverse it and
2450 ED537424	00820	LD (CKCHAR),DE	; put it in the filter
2454 C9	00830	RET	
	00840 ;**==*		
	00850 ;		Actual CLICK filter Code
	00860 ;**==*		
2455 180C	00870	HEADER JR FILTER	
2457 0000	00880	DW 0	;HIGH\$ before CLICK
2459 05	00890	DB 5,'CLICK'	
43 4C 49 43 4B			
245F 0000	00900	DCB DW \$-\$;DCB pointing to CLICK
2461 0000	00910	SPARE DW 0	;System wants it
	00920 ;		
	00930 ;		Is there a character here?
	00940 ;		
2463 DD2A5F24	00950	FILTER LD IX,(DCB)	;P/u DCB address
2467 3806	00960	JR C,NOTCTL	;Go if Get
2469 2804	00970	JR Z,NOTCTL	; or Put
246B	00980	IS_CTL @@CHNIO	;Pass the CTL call
246B 3E14	00007	LD A,20	
246D EF	00008	RST 40	
246E C9	00990	RET	
246F	01000	NOTCTL @@CHNIO	;Go to next in line
246F 3E14	00009	LD A,20	
2471 EF	00010	RST 40	
2472 C0	01010	RET NZ	;None - RETURN NZ
	01020 ;		
	01030 ;		Generate short Click
	01040 ;		
2473 F5	01050	SOUND PUSH AF	;Save registers
2474 0000	01060	CKCHAR DW 00	;Space for a CP instruct
2476 201C	01070	JR NZ,POPAF	; exit if CP above fails
2478 C5	01080	SNDNOW PUSH BC	
2479 D5	01090	PUSH DE	
	01100	IF @MOD2	
	01110	LD BC,LEN	;Duration

```

01120 LD A,-1 ;ON value
01130 OUT (SNDPORT),A ;Turn on sound
01140 LD A,16 ;Svc @PAUSE
01150 RST 28H ;Delay
01160 XOR A ;OFF value
01170 OUT (SNDPORT),A ;Turn off sound
01180 ENDIF
01190 ;
01200 IF @MOD4
01210 ;
247A 111848 01220 STFVALS LD DE,TONE<8!LEN ;D = Tone, E = Length
247D 3E00 01230 LD A,0 ;Init on/off toggle
247F 0E90 01240 LD C,SNDPORT ;Point to port
01250 ;
01260 ; ON portion
01270 ;
2481 3C 01280 DURLP INC A ;Hold output high
2482 ED79 01290 OUT (C),A ; for count of (B)
2484 42 01300 LD B,D ;Play tone
2485 10FE 01310 DJNZ $
01320 ;
01330 ;OFF portion
01340 ;
2487 3D 01350 DEC A ; for count of (B)
2488 ED79 01360 OUT (C),A
248A 42 01370 LD B,D ;Hold output low for
248B 10FE 01380 DJNZ $
01390 ;
248D 1D 01400 DEC E ;Dec the duration
248E 20F1 01410 JR NZ,DURLP
2490 10FE 01420 DJNZ $ ;Hold for 256 count
01430 ENDIF
01440 ;
2492 D1 01450 POP DE ;Restore regs
2493 C1 01460 POP BC
2494 F1 01470 POPAF POP AF
2495 C9 01480 RET ;And RETurn
01490 ;
0041 01500 LENGTH EQU $-HEADER ;Length of Filter
01510 ;
01520 ; INSTFLT - Relocate & Install Filter
01530 ;
2496 DD360047 01540 INSTFLT LD (IX+0),47H ;Set Filter,Ctl,Get,Put
01550 ;
01560 ; Pick up Old HIGH$ and save in driver
01570 ;
249A 210000 01580 LD HL,0 ;Get HIGH$
249D 45 01590 LD B,L
249E 01600 @@HIGH$
249E 3E64 00011 LD A,100
24A0 EF 00012 RST 40
24A1 225724 01610 LD (OLDHI),HL ;Stuff into header
01620 ;
01630 ; Calculate New HIGH$ & stuff into DCB
01640 ;
24A4 014100 01650 LD BC,LENGTH ;Length of driver
24A7 C5 01660 PUSH BC ;Save length
24A8 B7 01670 OR A
24A9 ED42 01680 SBC HL,BC ;HL => New HIGH$
24AB 01690 @@HIGH$ ;(B=0) set new HIGH$
24AB 3E64 00013 LD A,100

```

```

24AD EF      00014      RST      40
24AE 23      01700      INC      HL          ;Pt to driver
24AF DD7501   01710      LD       (IX+1),L    ;Stuff driver address
24B2 DD7402   01720      LD       (IX+2),H    ; into DCB
                01730 ;
                01740 ;      Calc offset between source & dest for relo
                01750 ;
24B5 115524   01760      LD       DE,HEADER ;Start of driver
24B8 E5       01770      PUSH     HL          ;Save Source & Dest ptrs
24B9 D5       01780      PUSH     DE
24BA B7       01790      OR       A          ;Clear carry
24BB ED52     01800      SBC      HL,DE       ;Get offset
                01810 ;
                01820 ;      Relocate internal references in driver
                01830 ;
24BD DD21DC24 01840      LD       IX,RELTBL   ;Point to relocation tbl
24C1 44       01850      LD       B,H        ;Move to BC
24C2 4D       01860      LD       C,L
24C3 DD6E00   01870      LD       L,(IX)      ;Get address to change
24C6 DD6601   01880      LD       H,(IX+1)
24C9 7C       01890      LD       A,H
24CA B5       01900      OR       L
24CB 2819     01910      JR       Z,RELDUN
24CD 5E       01920      LD       E,(HL)      ;P/U address
24CE 23       01930      INC      HL
24CF 56       01940      LD       D,(HL)
24D0 EB       01950      EX       DE,HL      ;Offset it
24D1 09       01960      ADD      HL,BC
24D2 EB       01970      EX       DE,HL
24D3 72       01980      LD       (HL),D     ;Put it back
24D4 2B       01990      DEC      HL
24D5 73       02000      LD       (HL),E
24D6 DD23     02010      INC      IX
24D8 DD23     02020      INC      IX
24DA 18E7     02030      JR       RLOOP     ;Loop till done
                02040 ;
                02050 ;      Relocation Table for Driver
                02060 ;
24DC 6524     02070      RELTBL  DW      FILTER+2,0,0,0,0
                0000 0000 0000 0000
                02080 ;
                02090 ;      Transfer Filter code to high memory
                02100 ;
24E6 E1       02110      RELDUN  POP      HL      ;HL => Source DE => Dest
24E7 D1       02120      POP      DE
24E8 C1       02130      POP      BC          ;BC = length of filter
24E9 EDB0     02140      LDIR          ;Block move
24EB C9       02150      RET              ;RETurn
                02160 ;
                02170 ;      DSPLY - Display a string
                02180 ;
24EC D5       02190      DSPLY   PUSH     DE      ;Save DE
24ED          02200      @@DSPLY ;Display it
                00015      IFEQ    00H,1
                00016      LD       HL,
                00017      ENDIF
24ED 3E0A     00018      LD       A,10
24EF EF       00019      RST      40
24F0 D1       02210      POP      DE
24F1 C8       02220      RET      Z          ;Return if good
                02230 ;

```

```

02240 ; IOERR - Any fatal Errors come here
02250 ;
24F2 6F 02260 IOERR LD L,A ;Xfer error # to HL
24F3 2600 02270 LD H,0 ;
24F5 F6C0 02280 OR 0C0H ;Short msg & RETURN
24F7 4F 02290 LD C,A
24F8 02300 @@ERROR ;Display error
24F8 3E1A 00020 LD A,26
24FA EF 00021 RST 40
24FB 180D 02310 JR EXIT ;Go to exit routine
02320 ;
02330 ; Error Handler
02340 ;
24FD 213C25 02350 VIASET LD HL,VIASET$ ;"Install with Set
2500 DD 02360 DB 0DDH
2501 212225 02370 CANT LD HL,CANT$ ;"No memory space
02380 ;
2504 02390 @@LOGOT ;Log error
00022 IFEQ 00H,1
00023 LD HL,
00024 ENDIF
2504 3E0C 00025 LD A,12
2506 EF 00026 RST 40
2507 21FFFF 02400 LD HL,-1 ;Set abort code
02410 ;
250A 310000 02420 EXIT LD SP,$-$ ;P/u original SP
250D 02430 @@CKBRKC ;Clear out break
250D 3E6A 00027 LD A,106
250F EF 00028 RST 40
2510 C9 02440 RET ; and RETURN
02450 ;
2511 43 02460 PRMTBL DB 'CHAR '
48 41 52 20 20
2517 4124 02470 DW CHARPRM+1
2519 43 02480 DB 'C '
20 20 20 20 20
251F 4124 02490 DW CHARPRM+1
2521 00 02500 NOP ;End of table
02510 ;
02520 ;
2522 4E 02530 CANT$ DB 'No memory space available',CR
6F 20 6D 65 6D 6F 72 79
20 73 70 61 63 65 20 61
76 61 69 6C 61 62 6C 65
0D
253C 4D 02540 VIASET$ DB 'Must install via SET',CR
75 73 74 20 69 6E 73 74
61 6C 6C 20 76 69 61 20
53 45 54 0D
02550 ;
2551 43 02560 HELLO$ DB 'CLICK'
4C 49 43 4B
2556 02570 *GET CLIENT:3
04240 ;CLIENTS/ASM - File to establish sign-on headers
04250 ;
2556 20 04260 DB ' - 6.2.0 - Copyright 1982/83/84 by Logical'
2D 20 36 2E 32 2E 30 20
2D 20 43 6F 70 79 72 69
67 68 74 20 31 39 38 32
2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61

```

```

        6C
2580 20      04270      DB      ' Systems, Inc.      ',10
        53 79 73 74 65 6D 73 2C
        20 49 6E 63 2E 20 20 20
        20 20 20 0A
        04280 ;
2595 41      04290      DB      'All Rights Reserved. Licensed 1982/83/84'
        6C 6C 20 52 69 67 68 74
        73 20 52 65 73 65 72 76
        65 64 2E 20 4C 69 63 65
        6E 73 65 64 20 31 39 38
        32 2F 38 33 2F 38 34
25BD 20      04300      DB      ' to xxxxxxxxxxxxxxxxxxx',10,13
        74 6F 20 78 78 78 78 78
        78 78 78 78 78 78 78 78
        78 78 78 78 78 0A 0D
        02580 ;
2400      02590      END      START

```


@@1	0000	@@2	0000	@@3	0000
@@4	0000	@MOD2	0000	@MOD4	FFFF
ABB	0010	AP	0027	BREAK	0080
BS	0008	CANT	2501	CANT\$	2522
CFLAG\$	0002	CHARPRM	2440	CKCHAR	2474
CR	000D	DCB	245F	DFLAG\$	0003
DOINIT	2417	DSPLY	24EC	DURLP	2481
ETX	0003	EXIT	250A	FILTER	2463
FLAG	0040	HEADER	2455	HELLO\$	2551
INSTFLT	2496	IOERR	24F2	IS_CTL	246B
KFLAG\$	000A	LEN	0018	LENGTH	0041
LF	000A	NORMEX	2413	NOTCTL	246F
NUM	0080	OLDHI	2457	PAR_ERR	002C
POPAF	2494	PRMTBL	2511	RELDUN	24E6
RELTBL	24DC	RLOOP	24C3	SFLAG\$	0012
SNDNOW	2478	SNDPORT	0090	SOUND	2473
SPARE	2461	START	2400	STARTA	2409
STFVALS	247A	STR	0020	TAB	0009
TOE	0048	VFLAG\$	0015	VIASET	24FD
VIASET\$	253C	@@ABORT	8020	@@ADTSK	80B3
@@BANK	85CB	@@BKSP	82AB	@@BREAK	85E1
@@CHNIO	800B	@@CKBRKC	862F	@@CKDRV	8107
@@CKEOF	82C0	@@CKTSK	809E	@@CLOSE	8296
@@CLS	8619	@@CMNDI	804A	@@CMNDR	805F
@@CTL	7E6F	@@DATE	7FE1	@@DCSTAT	8146
@@DEBUG	8089	@@DECHEX	854B	@@DIRRD	84B8
@@DIRWR	84CD	@@DIV16	8536	@@DIV8	8521
@@DODIR	811C	@@DSP	7E33	@@DSPLY	7ED3
@@ERROR	8074	@@EXIT	8035	@@FEXT	8425
@@FLAGS	85B5	@@FNAME	843A	@@FSPEC	8410
@@GATRD	84A3	@@GATWR	84E2	@@GET	7E47
@@GTDCB	8464	@@GTDCI	844F	@@GTMOD	8479
@@HDFMT	81EE	@@HEX16	858A	@@HEX8	8575
@@HEXDEC	8560	@@HIGH\$	859F	@@INIT	826C
@@KBD	7EAB	@@KEY	7E1F	@@KEYIN	7EBF
@@KLTSK	80F2	@@LOAD	83E6	@@LOC	82D5
@@LOF	82EA	@@LOGGER	7F0A	@@LOGOT	7F1F
@@MSG	7F56	@@MUL16	850C	@@MUL8	84F7
@@OPEN	8281	@@PARAM	7FCC	@@PAUSE	7FB7
@@PEOF	82FF	@@POSN	8314	@@PRINT	7F6B
@@PRT	7E83	@@PUT	7E5B	@@RAMDIR	8131
@@RDSEC	81C4	@@RDSSC	848E	@@READ	8329
@@REMOV	8257	@@RENAM	8242	@@REW	833E
@@RMTSK	80C8	@@RPTSK	80DD	@@RREAD	8353
@@RSLCT	81AF	@@RSTOR	8170	@@RUN	83FB
@@RWIT	8368	@@SEEK	819A	@@SEEKSC	837D
@@SKIP	8392	@@SLCT	815B	@@STEPI	8185
@@TIME	7FF6	@@VDCTL	7FA2	@@VER	83A7
@@VRSEC	81D9	@@WEOF	83BC	@@WHERE	7E97
@@WRITE	83D1	@@WRSEC	8203	@@WRSSC	8218
@@WTRK	822D				

2400 is the transfer address
00000 Total errors

NOTES:

NOTES:

COM/DVR - RS232 hardware driver

The Com driver program will initialize the UART and allow characters to be sent and received via the RS232 hardware. The driver will attempt to install itself in the low driver zone, but will relocate to high memory if necessary. It must be installed with the SET Library command.

```

00100 ;COM/ASM - RS232 Driver Program
0000 00110 TITLE '<COM/DVR - LS-DOS 6.2>'
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
00150 ;
0000 00160 *GET COPYCOM:3 ;Copyright message
00010 ; COPYCOM - File for Copyright COMMENT block
00020 ;
0000 00030 COM '<*(C) 1982,83,84 by LSI*>'
0000 00170 *GET SVCMAC:3 ;SVC Macro equivalents
00040 ;SVC MAC/ASM - LS-DOS Version VI
00050 *LIST OFF
03930 *LIST ON
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 BEGIN
2400 00220 @@CKBRKC
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,BEGINA ;Continue if no BREAK
2405 21FFFF 00240 LD HL,-1
2408 C9 00250 RET ;Return with abort code
00260 ;
2409 D5 00270 BEGINA PUSH DE ;Save DCB address
240A DDE1 00280 POP IX ; in index reg
240C ED537F26 00290 LD (CLDCB),DE ; and in driver header
2410 00300 @@DSPLY HELLO$ ;Welcome the user
00003 IFEQ 01H,1
2410 215B25 00004 LD HL,HELLO$
00005 ENDIF
2413 3E0A 00006 LD A,10
2415 EF 00007 RST 40
00310 ;
00320 ; Check if entry from SET command
00330 ;
2416 00340 @@FLAGS ;IY => flag table base
2416 3E65 00008 LD A,101
2418 EF 00009 RST 40
2419 FDCB025E 00350 BIT 3,(IY+'C'-'A') ;System request?
241D CA3325 00360 JP Z,VIASET ;"Install with Set
00370 ;
00380 ; Grab system dependent vectors
00390 ;
2420 FDE5 00400 PUSH IY ;Set DE to flag base
2422 D1 00410 POP DE
2423 210A00 00420 LD HL,'K'-'A' ;KFLAG$
2426 19 00430 ADD HL,DE
2427 223127 00440 LD (KFLAG),HL ;Save keyboard flag locn
242A 211200 00450 LD HL,'S'-'A' ;SFLAG$
242D 19 00460 ADD HL,DE
242E 225327 00470 LD (SFLAG),HL ;Save system flag location
2431 211600 00480 LD HL,'W'-'A' ;WRINT$
2434 19 00490 ADD HL,DE
2435 229026 00500 LD (WRINT),HL ;Save int mask
2438 21DEFF 00510 LD HL,10-44 ;INTVC$+10
243B 19 00520 ADD HL,DE ;Save for receive int
243C 228D26 00530 LD (INTVC),HL ; vector
00540 ;
00550 ; Move @ICNFG vector into driver

```

```

00560 ;
243F FD7E1C 00570 LD A,(IY+28) ;Get current opcode
2442 329A26 00580 LD (LINK),A ;Save in driver
2445 FD6E1D 00590 LD L,(IY+29) ;Get current address
2448 FD661E 00600 LD H,(IY+30) ;Put in driver code
244B 229B26 00610 LD (LINK+1),HL
00620 ;
00630 ; Check if driver already resident
00640 ;
244E 115725 00650 LD DE,CL$ ;Check if driver is
2451 00660 @@GTMOD ; already resident
2451 3E53 00670 LD A,83
2453 EF 00680 RST 40
2454 EB 00690 EX DE,HL ;Put DCB ptr to HL
2455 201A 006A0 JR NZ,NOTRES ;Go if not
006B0 ;
006C0 ; Make sure that the new DCB is same as the old
006D0 ;
2457 4E 006E0 LD C,(HL) ;P/u DCB pointer LSB
2458 23 006F0 INC HL
2459 46 00700 LD B,(HL) ;P/u DCB pointer MSB
245A 210600 00710 LD HL,6 ;Get old DCB name &
245D 09 00720 ADD HL,BC ; stuff into error
245E 7E 00730 LD A,(HL) ; message in case
245F 2C 00740 INC L ; a different DCB
2460 66 00750 LD H,(HL) ; is referenced
2461 6F 00760 LD L,A
2462 221D26 00770 LD (DCBNAM$),HL ;Stuff message with spec
2465 2A7F26 00780 LD HL,(CLDCB) ;P/u DCB existing DCB
2468 B7 00790 OR A ; pointer
2469 ED42 007A0 SBC HL,BC ;Same DCB pointer?
246B C23725 007B0 JP NZ,DCBERR ;Can't install if diff
246E C31425 007C0 JP ISRES
2471 114B49 007D0 NOTRES LD DE,'IK'
2474 007E0 @@GTDCB ;Locate low memory ptr
2474 3E52 007F0 LD A,82
2476 EF 00800 RST 40
2477 C24825 00810 JP NZ,IOERR ;Go if not found
247A 2D 00820 DEC L
247B 56 00830 LD D,(HL) ;P/u pointer to
247C 2D 00840 DEC L ; start of free
247D 5E 00850 LD E,(HL) ; low core
247E 22FA24 00860 LD (LCPTR+1),HL ;Save ptr for later
2481 21EF00 00870 LD HL,CLEND-CLDVR-1
2484 19 00880 ADD HL,DE ;Start + driver length
2485 22C424 00890 LD (SVEND+1),HL
2488 010013 008A0 LD BC,1300H ;Max addr + 1
248B AF 008B0 XOR A
248C ED42 008C0 SBC HL,BC ;See if room low
248E 382D 008D0 JR C,PUTLOW ; and install there if so
008E0 ;
008F0 ; Check if high memory available
00900 ;
2490 FDCB0246 00910 BIT 0,(IY+'C'-'A') ;Memory frozen?
2494 C23B25 00920 JP NZ,NOROOM ;Can't install if so
2497 210000 00930 LD HL,0
249A 45 00940 LD B,L ;Get HIGH$
249B 00950 @@HIGH$
249B 3E64 00960 LD A,100
249D EF 00970 RST 40
249E 22C424 00980 LD (SVEND+1),HL ;Top of driver

```

```

24A1 B7      01110      OR      A
24A2 01F000  01120      LD      BC,CLEND-CLDVR ; minus length
24A5 ED42    01130      SBC      HL,BC
24A7 0600    01140      LD      B,0
24A9 E5      01150      PUSH     HL
24AA        01160      @@HIGH$           ; is new HIGH$
24AA 3E64    00016      LD      A,100
24AC EF      00017      RST      40
24AD E1      01170      POP      HL
24AE 23      01180      INC      HL           ;Plus one is start
24AF 225525  01190      LD      (HCPTR),HL       ;Save it
24B2 215525  01200      LD      HL,HCPTR         ; and point to it
24B5 22FA24  01210      LD      (LCPTR+1),HL
24B8 3EFF    01220      LD      A,0FFH          ;Flag himem used
24BA 322525  01230      LD      (HGHFLG),A
                01240 ;
                01250 ; Relocate internal references in driver
                01260 ;
24BD DDE5    01270 PUTLOW PUSH     IX
24BF DD216727 01280      LD      IX,RELTAB         ;Point to relocation tbl
24C3 210000  01290 SVEND  LD      HL,$-$         ;Find distance to move
24C6 227926  01300      LD      (CLDVR+2),HL       ;Set last byte used
24C9 116627  01310      LD      DE,CLEND-1
24CC B7      01320      OR      A           ;Clear carry flag
24CD ED52    01330      SBC      HL,DE
24CF 44      01340      LD      B,H           ;Move to BC
24D0 4D      01350      LD      C,L
24D1 3E0D    01360      LD      A,TABLEN         ;Get table length
24D3 DD6E00  01370 RLOOP  LD      L,(IX)         ;Get address to change
24D6 DD6601  01380      LD      H,(IX+1)
24D9 5E      01390      LD      E,(HL)         ;P/U address
24DA 23      01400      INC      HL
24DB 56      01410      LD      D,(HL)
24DC EB      01420      EX      DE,HL         ;Offset it
24DD 09      01430      ADD      HL,BC
24DE EB      01440      EX      DE,HL
24DF 72      01450      LD      (HL),D         ;Put it back
24E0 2B      01460      DEC      HL
24E1 73      01470      LD      (HL),E
24E2 DD23    01480      INC      IX
24E4 DD23    01490      INC      IX
24E6 3D      01500      DEC      A
24E7 20EA    01510      JR      NZ,RLOOP         ;Loop till done
24E9 DDE1    01520      POP      IX         ;Restore DCB
                01530 ;
                01540 ; Set up @ICNFG
                01550 ;
24EB 218926  01560      LD      HL,INIT         ;Get (relocated)
24EC        01570 RX01  EQU      $-2
24EE FD751D  01580      LD      (IY+29),L         ; init address & put
24F1 FD741E  01590      LD      (IY+30),H         ; into system ICNFG area
24F4 3EC3    01600      LD      A,0C3H         ;Get JP instruction
24F6 FD771C  01610      LD      (IY+28),A         ;Turn on ICNFG
                01620 ;
                01630 ; Move driver
                01640 ;
24F9 210000  01650 LCPTR LD      HL,$-$         ;Low core or himem pointer
24FC 5E      01660      LD      E,(HL)
24FD 2C      01670      INC      L
24FE 56      01680      LD      D,(HL)
24FF D5      01690      PUSH     DE         ;Save start

```

```

2500 217726 01700 LD HL,CLDVR
2503 01F000 01710 LD BC,CLEND-CLDVR ;Calc driver length
2506 EDB0 01720 LDIR ;Move into place
2508 2AFA24 01730 LD HL,(LCPTR+1) ;If driver went low,
250B 73 01740 LD (HL),E ; need to update new
250C 2C 01750 INC L ; driver zone pointer
250D 72 01760 LD (HL),D
01770 ;
01780 ; Initialize the driver
01790 ;
250E F3 01800 DI
250F CD8926 01810 CALL INIT ;Init the UART
2510 01820 RX11 EQU $-2
2512 FB 01830 EI
01840 ;
2513 D1 01850 POP DE ;Pop filter start
01860 ;
2514 212026 01870 ISRES LD HL,CLACT$ ;Advise COM/DVR installed
2517 DD360007 01880 LD (IX),7 ;Init DCB type to "C/P/G"
251B DD7301 01890 LD (IX+1),E ; & stuff the driver
251E DD7202 01900 LD (IX+2),D ; address
2521 01910 @@LOGOT
00018 IFEQ 00H,1
00019 LD HL,
00020 ENDIF
2521 3E0C 00021 LD A,12
2523 EF 00022 RST 40
2524 3E00 01920 LD A,$-$ ;Did it use high memory?
2525 01930 HGHFLG EQU $-1
2526 B7 01940 OR A ;NZ if high
2527 2806 01950 JR Z,NTHGH
2529 215026 01960 LD HL,HMEM$ ;"Driver in himem..."
252C 01970 @@LOGOT
00023 IFEQ 00H,1
00024 LD HL,
00025 ENDIF
252C 3E0C 00026 LD A,12
252E EF 00027 RST 40
252F 210000 01980 NTHGH LD HL,0 ;Init on error code
2532 C9 01990 RET ; and exit
02000 ;
02010 ; Error exits
02020 ;
2533 213B26 02030 VIASET LD HL,VIASET$ ;"Install with Set
2536 DD 02040 DB 0DDH
2537 210126 02050 DCBERR LD HL,DCBERR$ ;"Driver being used already
253A DD 02060 DB 0DDH
253B 21E725 02070 NOROOM LD HL,NOROOM$ ;"Memory frozen
253E 02080 @@LOGOT
00028 IFEQ 00H,1
00029 LD HL,
00030 ENDIF
253E 3E0C 00031 LD A,12
2540 EF 00032 RST 40
2541 21FFFF 02090 LD HL,-1 ;Set abort code
2544 02100 @@CKBRKC ;Clear any break
2544 3E6A 00033 LD A,106
2546 EF 00034 RST 40
2547 C9 02110 RET
02120 ;
2548 6F 02130 IOERR LD L,A ;Error code to HL

```



```

2549 2600      02140      LD      H,0
254B F6C0      02150      OR      0C0H      ;Set short,return
254D 4F        02160      LD      C,A      ;Error to C
254E           02170      @@ERROR      ; for error dsply
254E 3E1A      00035      LD      A,26
2550 EF        00036      RST      40
2551           02180      @@CKBRKC      ;Clear any break
2551 3E6A      00037      LD      A,106
2553 EF        00038      RST      40
2554 C9        02190      RET
                02200      ;
                02210      ;      Messages & Data tables
                02220      ;
2555 0000      02230 HCPTR  DW      0      ;Save start if going to HIGH$
2557 24        02240 CL$   DB      '$CL',3
                43 4C 03
255B 52        02250 HELLO$ DB      'RS-232 Driver'
                53 2D 32 33 32 20 44 72
                69 76 65 72
                02260      ;
2568           02270 *GET   CLIENT:3
                03950 ;CLIENTS/ASM - File to establish sign-on headers
                03960 ;
2568 20        03970      DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
                2D 20 36 2E 32 2E 30 20
                2D 20 43 6F 70 79 72 69
                67 68 74 20 31 39 38 32
                2F 38 33 2F 38 34 20 62
                79 20 4C 6F 67 69 63 61
                6C
2592 20        03980      DB      ' Systems, Inc.      ',10
                53 79 73 74 65 6D 73 2C
                20 49 6E 63 2E 20 20 20
                20 20 20 0A
                03990      ;
25A7 41        04000      DB      'All Rights Reserved. Licensed 1982/83/84'
                6C 6C 20 52 69 67 68 74
                73 20 52 65 73 65 72 76
                65 64 2E 20 4C 69 63 65
                6E 73 65 64 20 31 39 38
                32 2F 38 33 2F 38 34
25CF 20        04010      DB      ' to xxxxxxxxxxxxxxxxxxx',10,13
                74 6F 20 78 78 78 78 78
                78 78 78 78 78 78 78 78
                78 78 78 78 78 0A 0D
                02280      ;
25E7 4E        02290 NOROOM$ DB      'No memory space available',CR
                6F 20 6D 65 6D 6F 72 79
                20 73 70 61 63 65 20 61
                76 61 69 6C 61 62 6C 65
                0D
2601 44        02300 DCBERR$ DB      'Driver already attached to *xx',CR
                72 69 76 65 72 20 61 6C
                72 65 61 64 79 20 61 74
                74 61 63 68 65 64 20 74
                6F 20 2A 78 78 0D
261D           02310 DCBNAM$ EQU      $-3
2620 43        02320 CLACT$ DB      'COM driver is now resident',CR
                4F 4D 20 64 72 69 76 65
                72 20 69 73 20 6E 6F 77
                20 72 65 73 69 64 65 6E

```

```

74 0D
263B 4D      02330 VIASET$ DB      'Must install via SET',CR
75 73 74 20 69 6E 73 74
61 6C 6C 20 76 69 61 20
53 45 54 0D
2650 0A      02340 HMEM$  DB      LF,'Note: driver installed in high memory',CR
4E 6F 74 65 3A 20 64 72
69 76 65 72 20 69 6E 73
74 61 6C 6C 65 64 20 69
6E 20 68 69 67 68 20 6D
65 6D 6F 72 79 0D
      02350 ;
00E0      02360 @WRINT EQU      0E0H
0080      02370 WRINT$ EQU      80H
      02380 ;
00E8      02390 MASRES EQU      0E8H      ;RS232 ports
00E8      02400 MODSTAT EQU      0E8H
00E9      02410 BAUDSET EQU      0E9H
00EA      02420 UARTCTL EQU      0EAH
00EA      02430 UARTST EQU      0EAH
00EB      02440 DATAREG EQU      0EBH
      02450 ;
      02460 ;      Actual driver
      02470 ;
2677      02480 CLDVR EQU      $
2677 1831    02490 JR      CLBGN      ;Branch around linkage
2679 6727    02500 DW      CLEND      ;Last byte used
267B 03      02510 DB      3,'$CL'
      24 43 4C
267F 0000    02520 CLDCB DW      $-$
2681 0000    02530 DW      0
2683      02540 CLDATA$ EQU      $
0000      02550 MSMASK EQU      $-CLDATA$
2683 00      02560 DB      0
      02570 ;
      02580 ;      UART control port image
      02590 ;
      02600 ;      bit 7: 1 = even parity, 0 = odd parity
      02610 ;      bits 6,5: word length <00=5, 10=6, 01=7, 11=8>
      02620 ;      bit 4: 1 = 2 stop bits, 0 = 1 stop bit
      02630 ;      bit 3: 1 = disable parity, 0 = enable parity
      02640 ;      bit 2: 1 = enable transmit data, 0 = break
      02650 ;      bit 1: 0 = Data Terminal Ready
      02660 ;      bit 0: 0 = Request to Send
      02670 ;
0001      02680 UCIMAGE EQU      $-CLDATA$
2684 A5      02690 DB      0A5H
2685 55      02700 BAUDRT DB      55H      ;Init 300 baud
0003      02710 LOGBRK EQU      $-CLDATA$
2686 03      02720 DB      3      ;Default is Control-C
0004      02730 CLFLG EQU      $-CLDATA$
2687 00      02740 DB      0      ;Init no char in buf
0005      02750 CLBUF EQU      $-CLDATA$
2688 00      02760 DB      0      ;One-char buffer
      02770 ;
      02780 ;      CL initialization routine. Set up DR interrupt
      02790 ;      vector & initialize the hardware
      02800 ;
2689 211827  02810 INIT LD      HL,RECVINT      ;Vector address
268A      02820 RX02 EQU      $-2
268C 220000  02830 LD      ($-),HL      ;INTVC$+10

```

```

268D      02840 INTVC EQU    $-2
268F 218000 02850 LD      HL,WRINT$ ;Interrupt enable mask
2690      02860 WRINT EQU    $-2
2692 CBEE    02870 SET     5,(HL) ;Enable RS232 DR
2694 7E      02880 LD      A,(HL)
2695 D3E0    02890 OUT     (@WRINT),A
2697 CD9D26 02900 CALL    CTL2 ;Init the hardware
2698      02910 RX03 EQU    $-2
269A C9      02920 LINK    RET     ;Link back thru any
269B 00      02930 DB      0,0 ; existing ICNFG
      00
      02940 ;
      02950 ; Initialize the UART & BRG
      02960 ;
269D ED4B8426 02970 CTL2 LD      BC,(CLDATA$+UCIMAGE) ;P/u values from DCB
269F      02980 RX04 EQU    $-2
26A1 D3E8    02990 OUT     (MASRES),A ;Reprime UART
26A3 79      03000 LD      A,C
26A4 D3EA    03010 OUT     (UARTCTL),A
26A6 78      03020 LD      A,B
26A7 D3E9    03030 OUT     (BAUDSET),A
26A9 C9      03040 RET
      03050 ;
26AA DD218326 03060 CLBGN LD      IX,CLDATA$ ;Point to data area
26AC      03070 RX05 EQU    $-2
26AE 3855    03080 JR      C,RECV ;Go if @GET request
26B0 2841    03090 JR      Z,SEND ;Go if @PUT request
26B2 79      03100 LD      A,C ;P/U @CTL byte
26B3 B7      03110 OR      A ;@CTL 00 ?
26B4 2826    03120 JR      Z,CANISND ;Go if so
26B6 3D      03130 DEC     A ;@CTL 01 ?
26B7 2857    03140 JR      Z,CTL1 ;Go if so
26B9 3D      03150 DEC     A ;Was it CTL-2 "INIT UART"
26BA 28E1    03160 JR      Z,CTL2 ;Go if so
26BC FE02    03170 CP      4-2 ;Wakeup feature?
26BE 2802    03180 JR      Z,CTL4 ;Go if wakeup feature
26C0 AF      03190 XOR     A
26C1 C9      03200 RET
      03210 ;
26C2 FDE5    03220 CTL4 PUSH    IY ;Transfer pointer to HL
26C4 E1      03230 POP     HL
26C5 7C      03240 LD      A,H ;Test if set or reset
26C6 B5      03250 OR      L
26C7 3EC9    03260 LD      A,0C9H ;Init disable wakeup
26C9 EB      03270 EX      DE,HL ;Switch new value to DE
26CA 2A2127 03280 LD      HL,(WAKEADR+1) ; & p/u old in HL
26CB      03290 RX06 EQU    $-2
26CD 2802    03300 JR      Z,SETWAK ;Jump if disable
26CF 3EC3    03310 LD      A,0C3H ;Make enable
26D1 322027 03320 SETWAK LD      (WAKEADR),A ;Load the opcode
26D2      03330 RX07 EQU    $-2
26D4 ED532127 03340 LD      (WAKEADR+1),DE ;Then the address
26D6      03350 RX08 EQU    $-2
26D8 E5      03360 PUSH    HL ;Transfer pointer to IY
26D9 FDE1    03370 POP     IY
26DB C9      03380 RET
      03390 ;
      03400 ; Check if ready to send
      03410 ;
26DC DBEA    03420 CANISND IN      A,(UARTST) ;Look at TX empty bit
26DE 2F      03430 CPL ;Flip it

```

The Source	UTILITY Files	COM/DVR - LS-DOS 6.2	Page 00008
26DF E640	03440	AND 40H	;Mask out all else
26E1 DBE8	03450	IN A,(MODSTAT)	;P/U modem status reg
26E3 C0	03460	RET NZ	;Return if can't send
26E4 47	03470	LD B,A	;Save modem status reg
26E5 DDAE00	03480	XOR (IX+MSMASK)	;Mask for which to flip
26E8 1F	03490	RRA	;Move into bits 0-3
26E9 1F	03500	RRA	
26EA 1F	03510	RRA	
26EB 1F	03520	RRA	
26EC DDA600	03530	AND (IX+MSMASK)	;Mask for which to check
26EF E60F	03540	AND 0FH	;Mask off garbage
26F1 78	03550	LD A,B	;Get back reg
26F2 C9	03560	RET	;Ret with Z or NZ
	03570 ;		
	03580 ;	Send character	
	03590 ;		
26F3 DD7E01	03600 SEND	LD A,(IX+UCIMAGE)	;Get UART ctrl reg
26F6 D3EA	03610	OUT (UARTCTL),A	;Put it (clears BREAK)
26F8 CDDC26	03620 SWAIT	CALL CANISND	;Poll
26F9	03630 RX09	EQU \$-2	
26FB 20FB	03640	JR NZ,SWAIT	; until ready
26FD 79	03650	LD A,C	;Get byte to send
26FE D3EB	03660	OUT (DATAREG),A	;Send it with Z-flag
2700 C9	03670	RET	; unchanged for return
	03680 ;		
	03690 ;	Receive character - Get from buffer if available	
	03700 ;		
2701 CD2327	03710 RECV1	CALL CKINP	;Ck if avail from port
2702	03720 RX10	EQU \$-2	
2704 C0	03730	RET NZ	;Back if none
2705 DDCB0426	03740 RECV	SLA (IX+CLFLG)	;Ck if avail from buf
2709 30F6	03750	JR NC,RECV1	;Go if none avail
270B DD7E05	03760	LD A,(IX+CLBUF)	;Get the char
270E BF	03770	CP A	;Set Z-flag & exit
270F C9	03780	RET	
	03790 ;		
	03800 ;	Break request	
	03810 ;		
2710 DD7E01	03820 CTL1	LD A,(IX+UCIMAGE)	;Pick up UART ctl image
2713 CB97	03830	RES 2,A	;Show BREAK bit
2715 D3EA	03840	OUT (UARTCTL),A	
2717 C9	03850	RET	;With Z-flag
	03860 ;		
	03870 ;	Data received interrupt handler	
	03880 ;		
2718 DD218326	03890 RECVINT	LD IX,CLDATA\$;Base of data area
271A	03900 RX13	EQU \$-2	
271C CD2327	03910	CALL CKINP	;See if available from port
271D	03920 RX12	EQU \$-2	
271F 78	03930	LD A,B	
2720 C9	03940 WAKEADR	RET	;Wakeup if enabled
2721 0000	03950	DW 0	;Space for address
	03960 ;		
	03970 ;	Routine to check on a received character	
	03980 ;		
2723 DBEA	03990 CKINP	IN A,(UARTST)	;Check if actually RX
2725 47	04000	LD B,A	;Save status
2726 E680	04010	AND 80H	;Mask Data Received bit
2728 EE80	04020	XOR 80H	;Set NZ if none avail
272A 3E00	04030	LD A,0	;Set "No error"
272C C0	04040	RET NZ	;Return if none

```

272D DBEB      04050      IN      A,(DATA REG)      ;Pick up character
272F 4F        04060      LD      C,A              ;Save tempy in reg-C
              04070 ;
              04080 ;      Break, Pause & Enter handler routine
              04090 ;
2730 210000    04100      LD      HL,$-$          ;KFLAG$
2731          04110 KFLAG EQU      $-2
2733 FE0D      04120      CP      CR              ;ENTER char received?
2735 2004      04130      JR      NZ,PAWSCK        ;Go if not
2737 CBD6      04140      SET     2,(HL)           ;Set ENTER bit
2739 1823      04150      JR      RECVEX
              04160 ;
273B FE60      04170 PAWSCK CP      60H           ;Pause char received?
273D 2004      04180      JR      NZ,BRKCHK        ;Go if not
273F CBCE      04190      SET     1,(HL)           ;Set pause bit
2741 181B      04200      JR      RECVEX
              04210 ;
2743 DD7E03    04220 BRKCHK LD      A,(IX+LOGBRK)   ;Break char received?
2746 B7        04230      OR      A              ;Check if LOGBRK=0
2747 2815      04240      JR      Z,RECVEX         ;No valid break if =0
2749 B9        04250      CP      C              ;Check if a valid BREAK
274A 2806      04260      JR      Z,BRKRECD        ;Go if so
274C DBEA      04270      IN      A,(UARTST)       ;Check for framing error
274E CB67      04280      BIT     4,A
2750 280C      04290      JR      Z,RECVEX         ;Quit if none
              04300 ;
              04310 ;      A BREAK was received, ck system's BREAK disable
              04320 ;
2752 3A0000    04330 BRKRECD LD      A,($-$)       ;Check if break key
2753          04340 SFLAG EQU      $-2
2755 E610      04350      AND     10H           ; is disabled
2757 3E00      04360      LD      A,0            ;Return NZ & A=0 if
2759 C0        04370      RET     NZ              ; the BREAK is disabled
275A CBC6      04380      SET     0,(HL)         ;Else set break bit
275C 0E80      04390      LD      C,80H         ; & reset BREAK code
275E DD7105    04400 RECVEX LD      (IX+CLBUF),C   ;Put char into 1-char buf
2761 DD360480  04410      LD      (IX+CLFLG),80H  ; & set char available
2765 AF        04420      XOR     A              ;Set Z flag
2766 C9        04430      RET
2767          04440 CLEND EQU      $
              04450 ;
2767 EC24      04460 RELTAB DW      RX01,RX02,RX03,RX04,RX05,RX06,RX07,RX08
              8A26 9826 9F26 AC26 CB26 D226 D626
2777 F926      04470      DW      RX09,RX10,RX11,RX12,RX13
              0227 1025 1D27 1A27
000D          04480 TABLEN EQU      $-RELTAB/2
              04490 ;
2400          04500      END      BEGIN

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
@WRINT	00E0 BAUDRT	2685 BAUDSET	00E9
BEGIN	2400 BEGINA	2409 BRKCHK	2743
BRKRECD	2752 CANISND	26DC CKINP	2723
CL\$	2557 CLACT\$	2620 CLBGN	26AA
CLBUF	0005 CLDATA\$	2683 CLDCB	267F
CLDVR	2677 CLEND	2767 CLFLG	0004
CR	000D CTL1	2710 CTL2	269D
CTL4	26C2 DATAREG	00EB DCBERR	2537
DCBERR\$	2601 DCBNAM\$	261D HCPTR	2555
HELLO\$	255B HGFLG	2525 HMEM\$	2650
INIT	2689 INTVC	268D IOERR	2548
ISRES	2514 KFLAG	2731 LCPTR	24F9
LF	000A LINK	269A LOGBRK	0003
MASRES	00E8 MODSTAT	00E8 MSMASK	0000
NOROOM	253B NOROOM\$	25E7 NOTRES	2471
NTHGH	252F PAWSCK	273B PUTLOW	24BD
RECV	2705 RECV1	2701 RECVEX	275E
RECVINT	2718 RELTAB	2767 RLOOP	24D3
RX01	24EC RX02	268A RX03	2698
RX04	269F RX05	26AC RX06	26CB
RX07	26D2 RX08	26D6 RX09	26F9
RX10	2702 RX11	2510 RX12	271D
RX13	271A SEND	26F3 SETWAK	26D1
SFLAG	2753 SVEND	24C3 SWAIT	26F8
TABLEN	000D UARTCTL	00EA UARTST	00EA
UC IMAGE	0001 VIASET	2533 VIASET\$	263B
WAKEADR	2720 WRINT	2690 WRINT\$	0080
@@ABORT	948A @@ADTSK	951D @@BANK	9A35
@@BKSP	9715 @@BREAK	9A4B @@CHNIO	9475
@@CKBRKC	9A99 @@CKDRV	9571 @@CKEOF	972A
@@CKTSK	9508 @@CLOSE	9700 @@CLS	9A83
@@CMNDI	94B4 @@CMNDR	94C9 @@CTL	92D9
@@DATE	944B @@DCSTAT	95B0 @@DEBUG	94F3
@@DECHEX	99B5 @@DIRRD	9922 @@DIRWR	9937
@@DIV16	99A0 @@DIV8	998B @@DODIR	9586
@@DSP	929D @@DSPLY	933D @@ERROR	94DE
@@EXIT	949F @@FEXT	988F @@FLAGS	9A1F
@@FNAME	98A4 @@FSPEC	987A @@GATRD	990D
@@GATWR	994C @@GET	92B1 @@GTDCB	98CE
@@GTDCT	98B9 @@GTMOD	98E3 @@HDFMT	9658
@@HEX16	99F4 @@HEX8	99DF @@HEXDEC	99CA
@@HIGH\$	9A09 @@INIT	96D6 @@KBD	9315
@@KEY	9289 @@KEYIN	9329 @@KLTSK	955C
@@LOAD	9850 @@LOC	973F @@LOF	9754
@@LOGGER	9374 @@LOGOT	9389 @@MSG	93C0
@@MUL16	9976 @@MUL8	9961 @@OPEN	96EB
@@PARAM	9436 @@PAUSE	9421 @@PEOF	9769
@@POSN	977E @@PRINT	93D5 @@PRT	92ED
@@PUT	92C5 @@RAMDIR	959B @@RDSEC	962E
@@RDSSC	98F8 @@READ	9793 @@REMOV	96C1
@@RENAM	96AC @@REW	97A8 @@RMTSK	9532
@@RPTSK	9547 @@RREAD	97BD @@RSLCT	9619
@@RSTOR	95DA @@RUN	9865 @@RWIT	97D2
@@SEEK	9604 @@SEEKSC	97E7 @@SKIP	97FC
@@SLCT	95C5 @@STEPI	95EF @@TIME	9460
@@VDCTL	940C @@VER	9811 @@VRSEC	9643
@@WEOF	9826 @@WHERE	9301 @@WRITE	983B
@@WRSEC	966D @@WRSSC	9682 @@WRTRK	9697

2400 is the transfer address
00000 Total errors

NOTES:

NOTES:

COMM/CMD - Terminal program with file send and receive

The Comm utility program acts as a terminal for communications work. Its features include file send and receive, and fully buffered device I/O (including printer spooling).

```

0000      00100 *GET      LCOMM
0000      00010 ;LCOMM/ASM - COMM Communications Program
0000      00020          TITLE  <COMM - LS-DOS 6.2>
0000      00030          SUBTTL  '<Program Code Section>'
0000      00040 ;
FFFF      00050 BUFRD   EQU    -1                ;Set true
0080      00060 BREAK   EQU    80H                ;Char fm keyboard
000A      00070 LF      EQU    10
000D      00080 CR      EQU    13
0013      00090 XOFF    EQU    'S'&1FH
0000      00100 ;
0000      00110 *GET      SVCMAC:3                ;SVC Macro equivalents
0000      00120 ;SVMAC/ASM - LS-DOS Version VI
0000      00130 *LIST    OFF
0000      04010 *LIST    ON
0000      04030 *GET      COPYCOM:3                ;Copyright messages
0000      04040 ; COPYCOM - File for Copyright COMMENT block
0000      04050 ;
0000      04060          COM      '<*(C) 1982,83,84 by LSI*>'
0000      04070 ;
3000      04080 BASE    EQU    3000H
3000      04090          ORG      BASE
0000      04100 ;
3000 210000 04110 $EXIT   LD      HL,0                ;Init no error
3003 310000 04120 QUIT$   LD      SP,$-$            ;P/u original stack
3004      04130 STACK    EQU    $-2
3006      04140          @@CKBRKC                ;Clear break bit
3006 3E6A    00001        LD      A,106
3008 EF      00002        RST      40
3009 C9      04150        RET
0000      04160 ;
300A 21FFFF 04170 $ABORT  LD      HL,-1            ;Set abort code
300D 18F4    04180        JR      QUIT$
0000      04190 ;
300F E5      04200 $OPEN  PUSH     HL
3010 210000 04210        LD      HL,$-$            ;Address of SFLAG$
3011      04220 SFLG     EQU    $-2
3013 CBC6    04230        SET     0,(HL)            ;Set open inhibit bit
3015 E1      04240        POP      HL
3016      04250          @@OPEN                    ;Do the open
3016 3E3B    00003        LD      A,59
3018 EF      00004        RST      40
3019 C9      04260        RET                        ;Return with status
0000      04270 ;
301A C5      04280 $ERROR  PUSH     BC
301B F6C0    04290        OR      0C0H                ;Set short,return
301D 4F      04300        LD      C,A                ;Error code to C
301E      04310          @@ERROR                    ; for error display
301E 3E1A    00005        LD      A,26
3020 EF      00006        RST      40
3021 C1      04320        POP      BC
3022 C9      04330        RET
0000      04340 ;
3023 3E00    04350 MAINLP LD      A,0                ;Test warning flag set
3025 B7      04360        OR      A                ; by OUTPUT on NEXTAP
3026 280F    04370        JR      Z,ENUFPG          ;Go if > 2K of space
3028 21AC37 04380        LD      HL,LILPG$          ;Display warning
302B      04390          @@DSPLY
0000      00007        IFEQ     00H,1
0000      00008        LD      HL,
0000      00009        ENDIF

```

Program Code Section

```

302B 3E0A      00010      LD      A,10
302D EF        00011      RST      40
302E 3E13      04400      LD      A,XOFF      ;Schedule a forced PUT
302F           04410 XOFFP2 EQU      $-1
3030 325234     04420      LD      (FRCPUT+1),A
3033 AF        04430      XOR      A
3034 322430     04440      LD      (MAINLP+1),A      ;Inhibit until next page
3037 DD219538   04450 ENUFPG LD      IX,KIVCTR      ;Get key from buffer if
303B CDB438     04460      CALL     PGMGET      ; available
303E 2022       04470      JR       NZ,SENDIT      ;Bypass if got one
3040 3E00       04480 FSSW   LD      A,0            ;FS On/Off (XMIT File)
3042 B7         04490      OR       A
3043 2832       04500      JR       Z,FSOFF      ;Bypass if not XMTG
3045 3AAD38     04510 CKFREPG LD      A,(FREEPG)      ;Don't get from file
3048 FE0C       04520      CP       12            ; if < 3K buffer space
304A DA7730     04530      JP       C,FSOFF      ;Go if less
304D 110438     04540      LD      DE,FS_FCB      ;Get sending FCB
3050           04550 FSSWGO @@GET      ;Get a byte to XMIT
3050 3E03       00012      LD      A,3
3052 EF        00013      RST      40
3053 280D       04560      JR       Z,SENDIT      ;Bypass if got byte
3055 FE1C       04570      CP       1CH          ;EOF encountered?
3057 2803       04580      JR       Z,EOFFS      ;Bypass if EOF
3059 CD1A30     04590      CALL     $ERROR      ;Output error message
305C CDDA33     04600 EOFFS  CALL     FS_OFF      ;Turn off XMIT
305F C3F430     04610      JP       SKIPREC      ; and ignore this round
3062 4F         04620 SENDIT LD      C,A          ;Xfer byte
3063 FE00       04630 XLTS1  CP       0            ;Single character send
3065 2002       04640      JR       NZ,DPLXSW      ; translate table
3067 0E00       04650 XLTS2  LD      C,0
3069 0600       04660 DPLXSW LD      B,0            ;Duplex On/Off
306B 04         04670      INC      B
306C 05         04680      DEC      B            ;Display on our devices
306D C41631     04690      CALL     NZ,DEVOUT      ; if duplex on (half)
3070 3AFA33     04700 LCMON  LD      A,(TASK8A+2)      ;Ck CL on
3073 B7         04710      OR       A
3074 C40231     04720      CALL     NZ,SNDOUT      ;Send char if ON
3077 3AFA33     04730 FSOFF  LD      A,(TASK8A+2)      ;Test for CL ON
307A B7         04740      OR       A
307B CAF430     04750      JP       Z,SKIPREC      ;Go if not
307E DD219D38   04760      LD      IX,CLREC
3082 CDB438     04770      CALL     PGMGET      ;Ck for char avail
3085 CAF430     04780      JP       Z,SKIPREC      ;Go if no char
3088 0600       04790 DSPCTRL LD      B,0          ;Ck if display of control
308A 04         04800      INC      B            ; codes is in effect
308B 05         04810      DEC      B
308C 2813       04820      JR       Z,SAVCHR      ;Go if no ctrl display
308E FE20       04830      CP       20H
3090 300F       04840      JR       NC,SAVCHR      ;Go if not ctrl
3092 F5         04850      PUSH     AF            ;Save the char
3093 21BF35     04860      LD      HL,BRAKET+1      ;Pt to control char msg
3096 4F         04870      LD      C,A
3097           04880      @@HEX8      ;Cvrt char & stuff in buf
3097 3E62       00014      LD      A,98
3099 EF        00015      RST      40
309A 21BE35     04890      LD      HL,BRAKET      ;Start of msg string
309D           04900      @@DSPLY      ;Display ASCII control value
309D           00016      IFEQ     00H,1
309D           00017      LD      HL,

```

Program Code Section

```

00018      ENDIF
309D 3E0A    00019      LD      A,10
309F EF      00020      RST     40
30A0 F1      04910      POP     AF          ;Rcvr char
30A1 4F      04920 SAVCHR LD      C,A          ;Save char
30A2 0600    04930 SHAKE LD      B,0          ;Handshake On/Off
30A4 04      04940      INC     B
30A5 05      04950      DEC     B
30A6 2820    04960      JR      Z,ECHOSW      ;Go if off
30A8 FE11    04970      CP      'Q'&1FH      ;Ctrl-Q?
30A9          04980 XONP1 EQU     $-1          ;Modify if PARM
30AA 2806    04990      JR      Z,CTLQ        ;Go if so
30AC FE13    05000      CP      'S'&1FH      ;Ctrl-S?
30AD          05010 XOFFP1 EQU     $-1        ;Modify if parm entered
30AE 2008    05020      JR      NZ,NOSQ       ;Go if neither
30B0 0600    05030      LD      B,0          ;Turn off
30B2 78      05040 CTLQ  LD      A,B          ; or on
30B3 324534  05050      LD      (TASK8B+1),A  ;*CL send task
30B6 183C    05060      JR      SKIPREC       ;Discard ctrl code
30B8 FE12    05070 NOSQ  CP      'R'&1FH      ;Ctrl-R?
30BA 2806    05080      JR      Z,CTLR        ;Go if so
30BC FE14    05090      CP      'T'&1FH      ;Ctrl-T?
30BE 2008    05100      JR      NZ,ECHOSW     ;Go if neither
30C0 0600    05110      LD      B,0          ;Turn off
30C2 78      05120 CTLR  LD      A,B          ; or on
30C3 322E31  05130      LD      (FRSW+1),A    ;FR device
30C6 182C    05140      JR      SKIPREC       ;Discard ctrl code
          05150 ;
          05160 ;      Test for ECHO after checking for handshake chars
          05170 ;
30C8 0600    05180 ECHOSW LD      B,0          ;Echo On/Off?
30CA 04      05190      INC     B
30CB 05      05200      DEC     B
30CC C4FA30  05210      CALL    NZ,CLOUT      ;Send char back if ON
30CF 79      05220      LD      A,C
30D0 FE0D    05230      CP      CR          ;Was it a CR?
30D2 200B    05240      JR      NZ,NOTCR
30D4 CD0931  05250      CALL    ECLF1        ;Send LF back if needed
30D7 21E130  05260      LD      HL,CRSW+1     ;Flag for CR recvd
          05270 ;
          05280 ;      Move state of ACCEPT LF switch into CRSW+1 when CR recv'd
          05290 ;
30DA 3E00    05300 ACCLFSW LD      A,0          ;Show CR found if accept
30DC 77      05310      LD      (HL),A        ; LF switch is off
30DD 1812    05320      JR      TAKEREC       ;Dsp CR
          05330 ;
          05340 ;      When LF rcv'd, delete if ACCLFSW is off & last char was CR
          05350 ;
30DF 79      05360 NOTCR LD      A,C          ;Check char
30E0 06FF    05370 CRSW  LD      B,0FFH      ;P/u del LF switch
30E2 21E130  05380      LD      HL,CRSW+1     ;Pt to switch
30E5 36FF    05390      LD      (HL),0FFH    ; (flip off switch -not CR)
30E7 FE0A    05400      CP      LF          ;Is line feed the char?
30E9 2006    05410      JR      NZ,TAKEREC   ;Go if not LF
30EB 3A1F34  05420      LD      A,(EIGHT+1)   ;Also skip if 8 bit
30EE B0      05430      OR      B          ; switch is off
30EF 2803    05440      JR      Z,SKIPREC     ;Skip LF if so
          05450 ;
30F1 CD1631  05460 TAKEREC CALL    DEVOUT      ;Out to active devices

```

Program Code Section

```

30F4 CDE933 05470 SKIPREC CALL TASKS ;Do 3 tasks (incl kbd)
30F7 C32330 05480 JP MAINLP ; & FRI0 test then loop
05490 ;
30FA 79 05500 CLOUT LD A,C ;Get char
30FB DD21A138 05510 LD IX,CSEND ;Set buffer pointers
30FF C3AE38 05520 JP OUTPGM ;Put in output buffer
05530 ;
3102 CDFA30 05540 SNDOUT CALL CLOUT ;Send this character
3105 79 05550 LD A,C ;Is it CR?
3106 FE0D 05560 CP CR
3108 C0 05570 RET NZ ;Done if not
05580 ;
3109 3E00 05590 ECLF1 LD A,$-$ ;Is echo linefeed on?
310A 05600 ECOLF EQU $-1
310B B7 05610 OR A
310C C8 05620 RET Z ;Done if not
310D 3E0A 05630 LD A,LF ;Otherwise load a LF
310F DD21A138 05640 LD IX,CSEND
3113 C3AE38 05650 JP OUTPGM ;Add to buffer/ret to caller
05660 ;
05670 ; Output to video
05680 ;
3116 3EFF 05690 DEVOUT LD A,0FFH ;Is *DO On/Off?
3118 B7 05700 OR A
3119 2812 05710 JR Z,FRSW ;Bypass if off
311B 79 05720 LD A,C
311C FE0C 05730 CP 0CH ;If formfeed,
311E 4F 05740 LD C,A
311F C5 05750 PUSH BC
3120 2007 05760 JR NZ,NOTCLS ; clear the screen
3122 0E1C 05770 LD C,1CH ;Cursor home
3124 05780 @@DSP
3124 3E02 00021 LD A,2
3126 EF 00022 RST 40
3127 0E1F 05790 LD C,1FH ;Clear to end-of-frame
3129 05800 NOTCLS @@DSP
3129 3E02 00023 LD A,2
312B EF 00024 RST 40
312C C1 05810 POP BC
05820 ;
05830 ; Send char to our disk if FR on
05840 ;
312D 3E00 05850 FRSW LD A,0 ;FR On/Off - receive file
312F B7 05860 OR A
3130 2808 05870 JR Z,PUTPR ;Bypass if FR off
3132 79 05880 LD A,C
3133 DD21A938 05890 LD IX,FRVCTR ;Put away into the
3137 CDAE38 05900 CALL OUTPGM ; FR buffer
05910 ;
05920 ; Place char into printer buffer if PR on
05930 ;
313A 3E00 05940 PUTPR LD A,0 ;PR On/Off?
313C B7 05950 OR A
313D 2808 05960 JR Z,FRIOSW ;Go if off
313F 79 05970 LD A,C
3140 DD219938 05980 LD IX,PRVCTR ;Place the char in
3144 CDAE38 05990 CALL OUTPGM ; the printer buffer
06000 ;
06010 ; Check if FR to disk is engaged

```

Program Code Section

```

06020 ;
3147 3EFF 06030 FRIOSW LD A,-1 ;Ck if FR-to-disk is on
3149 B7 06040 OR A
314A C8 06050 RET Z ;Go if not engaged
314B DD21A938 06060 LD IX,FRVCTR ;Is a char available
314F CDB438 06070 CALL PGMGET ; for the disk?
3152 C8 06080 RET Z ;Go if none for disk
3153 212438 06090 LD HL,FR_FCB ;Put char to disk
3156 CB7E 06100 BIT 7,(HL) ;OPEN FCB?
3158 C8 06110 RET Z ;Skip if not
3159 EB 06120 EX DE,HL
315A 4F 06130 LD C,A ;Place char in "C"
315B 06140 @@PUT ; and do the write
315B 3E04 00025 LD A,4
315D EF 00026 RST 40
315E C8 06150 RET Z ;Back if good
315F CD1A30 06160 CALL $ERROR
3162 CDE433 06170 CALL FRIO_OFF ;Turn FRI0 to disk off
3165 C3DF 33 06180 JP FR_OFF ;Turn FR off and return
06190 ;
06200 ; <CLEAR> command function entered - decode it
06210 ;
3168 010000 06220 CMDKEY LD BC,0 ;Init no device vector
316B 110000 06230 LD DE,0 ;Init no File FCB
316E 218930 06240 LD HL,DSPCTRL+1 ;Pt to ctrl char dsply parm
06250 IF @MOD4
3171 FEA7 06260 CP 27H!80H ;Display control chars?
06270 ENDIF
06280 IF @MOD2
06290 CP '&'+80H
06300 ENDIF
3173 CA3332 06310 JP Z,QFUNC
06320 ;
3176 216A30 06330 LD HL,DPLXSW+1
3179 FEA1 06340 CP '!' +80H ;Ck duplex
317B CA3332 06350 JP Z,QFUNC
06360 ;
317E 21C930 06370 LD HL,ECHOSW+1
06380 IF @MOD4
3181 FEA2 06390 CP ' "' +80H ;Ck echo
06400 ENDIF
06410 IF @MOD2
06420 CP '@'+80H
06430 ENDIF
3183 CA3332 06440 JP Z,QFUNC
06450 ;
3186 21A330 06460 LD HL,SHAKE+1 ;Check handshake
06470 IF @MOD4
3189 FEAA 06480 CP '*' +80H
06490 ENDIF
06500 IF @MOD2
06510 CP '_' +80H
06520 ENDIF
318B CA4232 06530 JP Z,QSHAKE
06540 ;
318E 210A31 06550 LD HL,ECOLF
3191 FEA3 06560 CP '#' +80H ;Echo line feed?
3193 CA3332 06570 JP Z,QFUNC
06580 ;

```

Program Code Section

```

3196 21DB30 06590 LD HL,ACCLFSW+1 ;Check accept-LF
3199 FEA4 06600 CP '$'+80H
319B CA3332 06610 JP Z,QFUNC
06620 ;
319E 211F34 06630 LD HL,EIGHT+1 ;Check 8-bit
06640 IF @MOD4
31A1 FEA9 06650 CP ')'+80H
06660 ENIF
06670 IF @MOD2
06680 CP '('+80H
06690 ENIF
31A3 CA3332 06700 JP Z,QFUNC
06710 ;
31A6 019538 06720 LD BC,KIVCTR ;Init *KI put/get index
31A9 218134 06730 LD HL,KISW+1
31AC FEB1 06740 CP '1'+80H ;CK *KI
31AE CA3332 06750 JP Z,QFUNC
06760 ;
31B1 010000 06770 LD BC,0 ;No *DO put/get index
31B4 211731 06780 LD HL,DEVOUT+1
31B7 FEB2 06790 CP '2'+80H ;CK *DO
31B9 2878 06800 JR Z,QFUNC
06810 ;
31BB 019938 06820 LD BC,PRVCTR ;Init *PR put/get index
31BE 213B31 06830 LD HL,PUTPR+1
31C1 FEB3 06840 CP '3'+80H ;CK *PR
31C3 286E 06850 JR Z,QFUNC
06860 ;
31C5 01A138 06870 LD BC,CLSEND ;Init *CL-S put/get index
31C8 21FA33 06880 LD HL,TASK8A+2
31CB FEB4 06890 CP '4'+80H ;CK *CL
31CD 2869 06900 JR Z,QCL
06910 ;
31CF 01A538 06920 LD BC,FSVCTR ;Init *FS put/get index
31D2 110438 06930 LD DE,FS_FCB ;Init *FS_FCB
31D5 DD21003A 06940 LD IX,XMTBUF ;Point to buffer
31D9 214130 06950 LD HL,FSSW+1
31DC FEB5 06960 CP '5'+80H ;CK FS
31DE 2853 06970 JR Z,QFUNC
06980 ;
31E0 01A938 06990 LD BC,FRVCTR ;P/u *FR put/get index
31E3 112438 07000 LD DE,FR_FCB ;P/u *FR_FCB
31E6 DD21003B 07010 LD IX,RCVBUF ;Pt to buffer
31EA 212E31 07020 LD HL,FRSW+1
31ED FEB6 07030 CP '6'+80H ;CK FR
31EF 2842 07040 JR Z,QFUNC
07050 ;
31F1 214831 07060 LD HL,FRIOSW+1
31F4 110000 07070 LD DE,0 ;No FCB here
31F7 FEB7 07080 CP '7'+80H ;Check FR IO to disk?
31F9 2838 07090 JR Z,QFUNC
07100 ;
31FB FEB8 07110 CP '8'+80H ;Menu request?
31FD CA9A32 07120 JP Z,MENU
07130 ;
07140 IF @MOD4
3200 FEA8 07150 CP '('+80H ;Local clear screen?
07160 ENIF
07170 IF @MOD2

```


Program Code Section

```

07180 CP      '*'+80H
07190 ENDIF
3202 CA8F32 07200 JP      Z,CLS
07210 ;
07220 IF      @MOD4
07230 CP      20H!80H      ;Clr-shf-0?
07240 ENDIF
07250 IF      @MOD2
07260 CP      ')+80H
07270 ENDIF
3207 CA6535 07280 JP      Z,DOSCMD      ;Do CMDR
07290 ;
07300 IF      @MOD4
07310 CP      '='+80H      ;CK LDOS exit
07320 ENDIF
07330 IF      @MOD2
07340 CP      '+'+80H
07350 ENDIF
320C C27D32 07360 JP      NZ,CMDERR
07370 ;
07380 ;
07390 ;      Exit from LCOMM - Remove task vectors
07400 ;
07410 EXIT
07420 IF      .NOT.BUFFRD
07430 LD      C,8      ;Remove comm line scan task
320F      07440 @@RMTSK
07450 LD      A,30
07460 RST     40
07470 ;
07480 LD      C,9      ;Rmv printer task if used
320F      07490 @@RMTSK
07500 LD      A,30
07510 RST     40
07520 ENDIF
07530 IF      BUFFRD
07540 LD      DE,CLDCB      ;Turn off wakeup feature
07550 LD      IY,$-$
320F 11E437 07560 EQU     $-2      ;Restoring previous state
3212 FD210000 07570 OLDVEC
3214      07580 LD      C,4
3216 0E04      07590 @@CTL
3218      07600 LD      A,5
3218 3E05      07610 RST     40
321A EF      07620 ENDIF
07630 ;
07640 CALL     FR OFF      ;Turn off any receive file
321B CDDF33 07650 LD      DE,FR FCB
321E 112438 07660 LD      A,(DE)
3221 1A      07670 BIT      7,A      ;Is it an open file?
3222 CB7F      07680 JP      Z,$EXIT      ;Exit if not else
3224 CA0030 07690 @@CLOSE      ; make sure it's closed
3227      07700 LD      A,60
3227 3E3C      07710 RST     40
3229 EF      07720 JP      Z,$EXIT      ;Exit if no error
322A CA0030 07730 CALL     $ERROR
322D CD1A30 07740 JP      $ABORT      ;Terminate
3230 C30A30 07750 ;
07760 ;      Query function ON or OFF
07770 ;

```

Program Code Section

```

07690 ;
3233 CD5A32 07700 QFUNC CALL QONOFF ;Get On or Off response
3236 77 07710 LD (HL),A ;Save which one
3237 C9 07720 RET
07730 ;
07740 ; Query *CL on or off
07750 ;
3238 CD5A32 07760 QCL CALL QONOFF
323B 77 07770 LD (HL),A
323C B7 07780 OR A ;On or off?
323D C8 07790 RET Z ;Quit if off
323E 324534 07800 LD (TASK8B+1),A ;Force CL-send on as well
3241 C9 07810 RET
07820 ;
07830 ; Query handshake on or off
07840 ;
3242 D5 07850 QSHAKE PUSH DE
3243 07860 @@KEY ;Get one key
3243 3E01 00035 LD A,1
3245 EF 00036 RST 40
3246 D1 07870 POP DE
3247 A7 07880 AND A ;Be sure flags are set
3248 FA5132 07890 JP M,QSHAKE1 ;Go if PF key
324B 326B34 07900 LD (AUTXOFF+1),A ;Save key as auto XOFF
324E 36FF 07910 LD (HL),0FFH ;Turn on handshake
3250 C9 07920 RET
3251 CD5F32 07930 QSHAKE1 CALL QONOFF1 ;Parse ON or OFF
3254 77 07940 LD (HL),A ;Turn on or off
3255 AF 07950 XOR A ;Turn off auto XOFF
3256 326B34 07960 LD (AUTXOFF+1),A
3259 C9 07970 RET
325A D5 07980 QONOFF PUSH DE ;Hang on to register
325B 07990 @@KEY ;Get the operand key
325B 3E01 00037 LD A,1
325D EF 00038 RST 40
325E D1 08000 POP DE ;Restore the register
325F 08010 QONOFF1 EQU $
08020 IF @MOD4
325F FEAD 08030 CP '-' + 80H ;Ck OFF
08040 ENDF
08050 IF @MOD2
08060 CP '=' + 80H
08070 ENDF
3261 2821 08080 JR Z,TURNOF ; and go if off
08090 IF @MOD4
3263 FEBA 08100 CP ':' + 80H ;Ck ON
08110 ENDF
08120 IF @MOD2
08130 CP '-' + 80H
08140 ENDF
3265 281F 08150 JR Z,TURNON ; and go if on
3267 E3 08160 POPERR EX (SP),HL ;Discard ret address
3268 E1 08170 POP HL
3269 FEB9 08180 CP '9' + 80H ;Ck ID
326B CA9633 08190 JP Z,FILID
08200 ;
326E FEB0 08210 CP '0' + 80H ;Ck RESET
3270 CA4C33 08220 JP Z,FILRES
08230 ;

```

Program Code Section

```

3273 FEA5      08240 CP      '%'+80H      ;Ck REWIND
3275 CA8433    08250 JP      Z,FILREW
               08260 ;
               08270 IF      @MOD4
3278 FEA6      08280 CP      '&'+80H      ;Ck PEOF
               08290 ENDF
               08300 IF      @MOD2
               08310 CP      '^'+80H
               08320 ENDF
327A CA8D33    08330 JP      Z,FILEOF
               08340 ;
327D 218D37    08350 CMDERR LD      HL,CMDERR$      ;None of above, dsply
3280           08360 @@DSPLY      ; "Unacceptable command..."
               08370 IFEQ     00H,1
               08380 LD      HL,
               08390 ENDF
3280 3E0A      08400 LD      A,10
3282 EF        08410 RST      40
3283 C9        08420 RET
               08430 ;
               08440 ;      Process OFF
               08450 ;
3284 AF        08460 TURNOF XOR      A      ;Off = 0
3285 C9        08470 RET
               08480 ;
               08490 ;      Process ON
               08500 ;
3286 EB        08510 TURNON EX      DE,HL      ;Shift "FCB" to HL
3287 CB7E      08520 BIT      7,(HL)      ;FCB on or non-file?
3289 EB        08530 EX      DE,HL      ;If non-file, HL now
328A 3EFF      08540 LD      A,0FFH      ; points to X'0000'
328C C0        08550 RET      NZ      ; which contains X'F3'
328D 18D8      08560 JR      POPERR      ;Is an error
               08570 ;
               08580 ;      Process Clear Screen
               08590 ;
328F 0E1C      08600 CLS      LD      C,1CH      ;Cursor home
3291           08610 @@DSP
3291 3E02      08620 LD      A,2
3293 EF        08630 RST      40
3294 0E1F      08640 LD      C,1FH      ;Clear to end-of-frame
3296           08650 @@DSP
3296 3E02      08660 LD      A,2
3298 EF        08670 RST      40
3299 C9        08680 RET
               08690 ;
               08700 ;      Process MENU
               08710 ;
329A           08720 MENU      EQU      $
329A 21E635    08730 LD      HL,STAT1      ;Clear top row status
329D 11E735    08740 LD      DE,STAT1+1      ;1st char always a space
32A0 014200    08750 LD      BC,66
32A3 EDB0      08760 LDIR
32A5 210537    08770 LD      HL,STAT2      ;Clear bottom row status
32A8 110637    08780 LD      DE,STAT2+1
32AB 0E26      08790 LD      C,38
32AD EDB0      08800 LDIR
32AF 060F      08810 LD      B,15      ;Init loop count
32B1 212D37    08820 LD      HL,STATAB      ;Words where status stored

```

Program Code Section

```

32B4 5E      08740 STATLP1 LD      E,(HL)          ;P/u lo-switch
32B5 23      08750          INC      HL
32B6 56      08760          LD      D,(HL)          ;P/u hi-switch
32B7 23      08770          INC      HL
32B8 7E      08780          LD      A,(HL)          ;P/u lo-stuff
32B9 23      08790          INC      HL
32BA E5      08800          PUSH     HL              ;Save pointer
32BB 66      08810          LD      H,(HL)          ;P/u hi-stuff
32BC 6F      08820          LD      L,A              ;Xfer lo-stuff
32BD 1A      08830          LD      A,(DE)          ;Get status
32BE B7      08840          OR       A              ;Active or not?
32BF 2802    08850          JR      Z,$+4          ;Go if not
32C1 362A    08860          LD      (HL),'*'        ; else stuf an '*'
32C3 E1      08870          POP      HL              ;Rcvr pointer
32C4 23      08880          INC      HL              ;Bump to next pos
32C5 10ED    08890          DJNZ     STATLP1
32C7 1A      08900          LD      A,(DE)          ;P/u shake again
32C8 B7      08910          OR       A
32C9 280D    08920          JR      Z,STATLP2        ;Go if off
32CB 3A6B34  08930          LD      A,(AUTXOFF+1)    ;Check if xoff char set
32CE B7      08940          OR       A
32CF 2807    08950          JR      Z,STATLP2        ;Skip if not special char
32D1 212536  08960          LD      HL,STAT1+63      ;Auto x-off char posn
32D4 4F      08970          LD      C,A
32D5         08980          @@HEX8
32D5 3E62    09048          LD      A,98
32D7 EF      09049          RST      40
32D8 21E535  08990 STATLP2 LD      HL,MNUMSG        ;Ptr to Comm menu
32DB         09000          @@DSPLY
32DB         09050          IFEQ     00H,1
32DB         09051          LD      HL,
32DB         09052          ENDIF
32DB 3E0A    09053          LD      A,10
32DD EF      09054          RST      40
32DE 210438  09010          LD      HL,FS_FCB        ;FS file open?
32E1 CB7E    09020          BIT      7,(HL)
32E3 281C    09030          JR      Z,STATLP3        ;Go if closed
32E5 114438  09040          LD      DE,DUMMY        ;Recover its name w/o
32E8 D5      09050          PUSH     DE              ; changing the FCB
32E9 012000  09060          LD      BC,32
32EC EDB0    09070          LDIR
32EE D1      09080          POP      DE
32EF ED4B4A38 09090          LD      BC,(DUMMY+6)    ;Get drive and DEC
32F3 D5      09100          PUSH     DE
32F4         09110          @@FNAME
32F4 3E50    09055          LD      A,80
32F6 EF      09056          RST      40
32F7 216937  09120          LD      HL,FSNAME$      ;Output "FS-SPEC: "
32FA         09130          @@DSPLY
32FA         09057          IFEQ     00H,1
32FA         09058          LD      HL,
32FA         09059          ENDIF
32FA 3E0A    09060          LD      A,10
32FC EF      09061          RST      40
32FD E1      09140          POP      HL              ;Rcvr fcb pointer and
32FE         09150          @@DSPLY        ; display the filespec
32FE         09062          IFEQ     00H,1
32FE         09063          LD      HL,
32FE         09064          ENDIF

```

Program Code Section

```

32FE 3E0A 00065 LD A,10
3300 EF 00066 RST 40
3301 212438 09160 STATLP3 LD HL,FR FCB ;Is the FR file open?
3304 CB7E 09170 BIT 7,(HL)
3306 281C 09180 JR Z,STATLP4 ;Go if closed
3308 114438 09190 LD DE,DUMMY ;Similar to above
330B D5 09200 PUSH DE
330C 012000 09210 LD BC,32
330F EDB0 09220 LDIR ;Create a duplicate FCB
3311 D1 09230 POP DE
3312 ED4B4A38 09240 LD BC,(DUMMY+6) ;P/u Drive & DEC
3316 D5 09250 PUSH DE
3317 09260 @@FNAME ;Call for name recover
3317 3E50 00067 LD A,80
3319 EF 00068 RST 40
331A 217337 09270 LD HL,FRNAME$ ;"FR-SPEC:"
331D 09280 @@DSPLY
00069 IFEQ 00H,1
00070 LD HL,
00071 ENDIF
331D 3E0A 00072 LD A,10
331F EF 00073 RST 40
3320 E1 09290 POP HL ;P/u name start
3321 09300 @@DSPLY ; and dsply it
00074 IFEQ 00H,1
00075 LD HL,
00076 ENDIF
3321 3E0A 00077 LD A,10
3323 EF 00078 RST 40
3324 3AAD38 09310 STATLP4 LD A,(FREEPG) ;How much buffer left
3327 0F 09320 RRCA ;Divide by 4 to show
3328 0F 09330 RRCA ; in K
3329 E63F 09340 AND 3FH ;No bit 7 nor 6
332B 218937 09350 LD HL,PAGSPR$+10 ;Where to stuff
332E 06FF 09360 LD B,-1 ;Init to count 10's
3330 04 09370 CVD1 INC B
3331 D60A 09380 SUB 10 ;How many tens?
3333 30FB 09390 JR NC,CVD1 ;Go if no more
3335 F5 09400 CVD2 PUSH AF ;Save remainder
3336 78 09410 LD A,B ;P/u tens
3337 B7 09420 OR A ;Was it zero tens?
3338 0620 09430 LD B,' ' ;Init for space
333A 2802 09440 JR Z,$+4 ;Go if no tens
333C 0630 09450 LD B,'0' ;Init for ASCII
333E 80 09460 ADD A,B ;Convert to ASCII
333F 77 09470 LD (HL),A ;Stuff & bump
3340 23 09480 INC HL
3341 F1 09490 POP AF ;Get remainder
3342 C63A 09500 ADD A,3AH ;Adjust units place
3344 77 09510 LD (HL),A
3345 217F37 09520 LD HL,PAGSPR$ ;"Memory: K"
3348 09530 @@DSPLY
00079 IFEQ 00H,1
00080 LD HL,
00081 ENDIF
3348 3E0A 00082 LD A,10
334A EF 00083 RST 40
334B C9 09540 RET
09550 ;

```

Program Code Section

```

09560 ;      Process RESET of a "device"
09570 ;
334C 78 09580 FILRES LD      A,B      ;Check if a device vector
334D B1 09590      OR      C          ; was passed
334E CA7D32 09600      JP      Z,CMDERR ;Go if not - is error
3351 7A 09610      LD      A,D      ;Check for a possible
3352 B3 09620      OR      E          ; FCB for disk
3353 2016 09630      JR      NZ,FILR4 ;Go if disk else device
09640 ;
09650 ;      Reset the page buffer(s) for the device
09660 ;
3355 F3 09670 FILR1 DI          ;No interrupts until done
3356 60 09680      LD      H,B      ;Xfer vector table entry
3357 69 09690      LD      L,C      ; to grab put/get index
3358 4E 09700      LD      C,(HL)   ;P/u the PUT pointer
3359 23 09710      INC     HL        ; and make the GET
335A 46 09720      LD      B,(HL)   ; pointer equal so
335B 23 09730      INC     HL        ; buffer contents show
335C 71 09740      LD      (HL),C   ; as empty
335D 23 09750      INC     HL
335E 7E 09760      LD      A,(HL)   ;P/u the GET pointer to
335F 70 09770      LD      (HL),B   ; check if in same page
3360 B8 09780 FILR2 CP      B      ;Is put/get in same page?
3361 2806 09790      JR      Z,FILR3 ;Go if it is
3363 67 09800      LD      H,A      ; else set up to free this
3364 CD4E35 09810      CALL    FNPIU  ; page by finding next
3367 18F7 09820      JR      FILR2  ;Loop until next = 1st
3369 FB 09830 FILR3 EI          ;Interrupts back on
336A C9 09840      RET
09850 ;
09860 ;      Reset a file device
09870 ;
336B 212438 09880 FILR4 LD      HL,FR_FCB ;Turn off the FR or FS
336E AF 09890      XOR      A
336F ED52 09900      SBC     HL,DE   ;Is this the FR?
3371 214130 09910      LD      HL,FSSW+1
3374 2006 09920      JR      NZ,OFFS
3376 324831 09930      LD      (FRIOSW+1),A ;Turn off FR IO to disk
3379 212E31 09940      LD      HL,FRSW+1 ;Turn off FR to buffer
337C 77 09950 OFFS LD      (HL),A   ;Turn off FR or FS
337D 09960      @@CLOSE ;Close the file
337D 3E3C 00084      LD      A,60
337F EF 00085      RST      40
3380 C41A30 09970      CALL    NZ,$ERROR ;Show any close error
3383 C9 09980      RET
09990 ;
10000 ;      Process REWIND
10010 ;
3384 7A 10020 FILREW LD      A,D      ;Rewind the specified
3385 B3 10030      OR      E          ; file (FCB given) if
3386 CA7D32 10040      JP      Z,CMDERR ; it is in use
3389 10050      @@REW
3389 3E44 00086      LD      A,68
338B EF 00087      RST      40
338C C9 10060      RET
10070 ;
10080 ;      Process PEOF
10090 ;
338D 7A 10100 FILEOF LD      A,D      ;Check if a file device

```

Program Code Section

```

338E B3      10110      OR      E      ; was specified
338F CA7D32  10120      JP      Z,CMDERR ;Go if not - is error
3392         10130      @@PEOF      ; else position to end
3392 3E41     00088      LD      A,65
3394 EF      00089      RST      40
3395 C9      10140      RET
          10150 ;
          10160 ;      Process ID request
          10170 ;
3396 7A      10180 FILID LD      A,D      ;Bad command if not
3397 B3      10190      OR      E      ; FS or FR specified
3398 CA7D32  10200      JP      Z,CMDERR ;Go on error
339B 1A      10210      LD      A,(DE)    ;Make sure that it is
339C 07      10220      RLCA      ; not already open
339D 3834    10230      JR      C,NOTNOW ;CF=already open
339F D5      10240      PUSH     DE
33A0 DDE5    10250      PUSH     IX      ;Save buffer pointer
33A2 21C335  10260      LD      HL,FILEPMT ;Prompt for filespec
33A5         10270      @@DSPLY
          00090      IFEQ     00H,1
          00091      LD      HL,
          00092      ENDIF
33A5 3E0A    00093      LD      A,10
33A7 EF      00094      RST      40
33A8 E1      10280      POP      HL      ;Take file name
33A9 01001F  10290      LD      BC,31<8  ;31 chars max
33AC         10300      @@KEYIN      ;Get the filespec
33AC 3E09    00095      LD      A,9
33AE EF      00096      RST      40
33AF F5      10310      PUSH     AF      ;Save flag state
33B0 0E0E    10320      LD      C,0EH    ;Turn the cursor back on
33B2         10330      @@DSP
33B2 3E02    00097      LD      A,2
33B4 EF      00098      RST      40
33B5 F1      10340      POP      AF      ;Rcvr KEYIN exit state
33B6 D1      10350      POP      DE
33B7 D8      10360      RET      C      ;Ret if BREAK from KEYIN
33B8 E5      10370      PUSH     HL      ;Save ptr to buffer
33B9         10380      @@FSPEC      ;Fetch & parse filespec
33B9 3E4E    00099      LD      A,78
33BB EF      00100      RST      40
33BC 210438  10390      LD      HL,FS_FCB ;Ck if FILID req from
33BF AF      10400      XOR      A      ; FS or FR
33C0 ED52    10410      SBC      HL,DE    ;What's the FCB?
33C2 E1      10420      POP      HL      ;Recover buffer
33C3 0600    10430      LD      B,0      ;LRL=256
33C5 2005    10440      JR      NZ,FILFR ;Go if req from FR
33C7 CD0F30  10450      CALL     $OPEN    ;Only OPEN a FS
33CA 1803    10460      JR      $+5      ;Branch around INIT
33CC         10470 FILFR @@INIT      ;Open the receive file
33CC 3E3A    00101      LD      A,58
33CE EF      00102      RST      40
33CF C41A30  10480      CALL     NZ,$ERROR ;Show any open error
33D2 C9      10490      RET
          10500 ;
33D3 21D135  10510 NOTNOW LD      HL,OPENMSG ;"File already open"
33D6         10520      @@DSPLY      ;Show why ID failed
          00103      IFEQ     00H,1
          00104      LD      HL,

```

Program Code Section

```

00105      ENDIF
33D6 3E0A    00106      LD      A,10
33D8 EF      00107      RST     40
33D9 C9      10530      RET
              10540 ;
              10550 ;      Routines to turn off file devices
              10560 ;
33DA AF      10570 FS_OFF XOR     A           ;File send
33DB 324130  10580      LD      (FSSW+1),A
33DE C9      10590      RET
33DF AF      10600 FR_OFF XOR     A           ;File receive
33E0 322E31  10610      LD      (FRSW+1),A
33E3 C9      10620      RET
33E4 AF      10630 FRIO_OFF XOR     A         ;Dump to disk
33E5 324831  10640      LD      (FRIOSW+1),A
33E8 C9      10650      RET
              10660 ;
              10670 ;      Call various tasks (on each main loop)
              10680 ;
33E9 F3      10690 TASKS DI
              10700 ;
              10710      IF      .NOT.BUFFRD      ;W fcn does this if bfrd
              10720      CALL    TASK8A           ;Try to receive from *CL
              10730      ENDIF
              10740 ;
33EA CD4434  10750      CALL    TASK8B           ;Try to send to *CL
33ED FB      10760      EI
33EE CD7434  10770      CALL    TASKK           ;Allow interrupts here
              10780      IF      .NOT.BUFFRD
              10790      DI                     ;If RS232 does not interrupt
              10800      ENDIF                  ;Printer must be task
33F1 CDB834  10810      CALL    TASK9
33F4 FB      10820      EI
33F5 C34731  10830      JP      FRIOSW           ;Check on dump to disk
              10840 ;
              10850 ;      INTERRUPT TASK 8
              10860 ; WO/buffer      A is done once per main loop + int rate
              10870 ;                      B is done once per main loop + int rate
              10880 ; W/buffer      A is done by wakeup feature + int rate
              10890 ;                      B is done once per main loop + int rate
              10900 ;
              10910      IF      .NOT.BUFFRD
              10920 TCB8      DW      TASK8
              10930 TASK8     CALL    TASK8A
              10940      JP      TASK8B
              10950      ENDIF
              10960 ;
33F8 F3      10970 TASK8A DI
33F9 3EFF    10980      LD      A,0FFH           ;CL recv On/Off
33FB B7      10990      OR      A
33FC C8      11000      RET     Z               ;Done if CL recv off
              11010 ;
              11020 ;      @GET handler to keep interrupts off if possible
              11030 ;
33FD 11E437  11040      LD      DE,CLDCB         ;=> OPEN DCB
3400 62      11050 FNDDVR LD      H,D           ;Xfer to HL
3401 6B      11060      LD      L,E
3402 7E      11070      LD      A,(HL)         ;Get DCB type
3403 CB6F    11080      BIT     5,A            ;Is it linked?

```


Program Code Section

```

3405 2013    11090    JR      NZ,LNKD      ;Need CHNIO if so
3407 23      11100    INC      HL          ;=>address field of DCB
3408 5E      11110    LD       E,(HL)      ;If routed, address is
3409 23      11120    INC      HL          ; next DCB to use
340A 56      11130    LD       D,(HL)      ; else EP of driver
340B CB67    11140    BIT      4,A        ;Z = not routed
340D 20F1    11150    JR      NZ,FNDDVR   ;Loop till not routed
340F E608    11160    AND      00001000B  ;Can't talk to NIL device
3411 C0      11170    RET      NZ
3412 EB      11180    EX       DE,HL      ;Address to HL
3413 111D34  11190    LD       DE,RETADD   ;Put RET address on stack
3416 D5      11200    PUSH     DE
3417 FE02    11210    CP       2          ;Set C,NZ for input request
3419 E9      11220    JP       (HL)       ;Go to driver
          11230 ;
341A        11240    LNKD    @@GET        ;Use SVC if LINKED
341A 3E03    00108    LD       A,3
341C EF      00109    RST      40
341D C0      11250    RETADD  RET      NZ      ;NZ means no char rcv'd
          11260 ;
341E 0600    11270    EIGHT  LD       B,0    ;Eight bit mode switch
3420 04      11280    INC      B
3421 05      11290    DEC      B
3422 2006    11300    JR      NZ,XLTR1    ;Go if 8 bit
3424 E67F    11310    AND      7FH        ;Strip bit 7
3426 C8      11320    RET      Z          ;Always ignore nulls
3427 FE7F    11330    CP       7FH        ; & DELETE if not 8-bit
3429 C8      11340    RET      Z
          11350 ;
          11360 ;      Do XLATER after stripping high bit
          11370 ;
342A FE00    11380    XLTR1   CP       $-$    ;Character to translate?
342C 2002    11390    JR      NZ,TSTNUL   ;Go if not a match
342E 3E00    11400    XLTR2   LD       A,$-$  ;Replace with xlated char
          11410 ;
          11420 ;      NULL Parm now only affects 8-bit mode
          11430 ;
3430 B7      11440    TSTNUL  OR       A      ;Is char a null?
3431 2005    11450    JR      NZ,KEEPCH   ;Go if not
3433 06FF    11460    ACCNUL  LD       B,0FFH ;Default to accept nulls
3435 04      11470    INC      B          ;NZ=nulls wanted
3436 05      11480    DEC      B          ;Z=don't accept nulls
3437 C8      11490    RET      Z
          11500 ;
3438 DDE5    11510    KEEPCH  PUSH     IX      ;Place in CL input buf
343A DD219D38 11520    LD       IX,CLREC
343E CDD634  11530    CALL     OUTPUT    ;Out to the buffer if
3441 DDE1    11540    POP      IX          ; non-null or want nulls
3443 C9      11550    RET
          11560 ;
3444 3EFF    11570    TASK8B  LD       A,0FFH ;CL send On/Off for
3446 B7      11580    OR       A          ; handshaking
3447 C8      11590    RET      Z
3448 0E00    11600    LD       C,0
344A 11E437  11610    LD       DE,CLDCB    ;Now xmit a CTL0 to
344D        11620    @@CTL                ; CL
344D 3E05    00110    LD       A,5
344F EF      00111    RST      40
3450 C0      11630    RET      NZ          ;Indicates not ready

```

Program Code Section

```

3451 0E00      11640 FRCPUT LD      C,$-$      ;Force a char out?
3453 AF        11650      XOR      A          ;Clear it after p/u
3454 325234    11660      LD      (FRCPUT+1),A
3457 B1        11670      OR       C          ;Check original status
3458 2000D     11680      JR       NZ,FRCIT   ;Go if force on
345A DDE5      11690      PUSH     IX
345C DD21A138  11700      LD      IX,CLSEND   ;Do we have a char to
3460 CD1635    11710      CALL     BUFGET    ; send to the CL?
3463 DDE1      11720      POP      IX
3465 C8        11730      RET       Z          ;RET if not
3466 4F        11740      LD      C,A        ;Save character
3467           11750 FRCPUT @@PUT          ;Put it out
3467 3E04      00112      LD      A,4
3469 EF        00113      RST      40
           11760 ;
346A 3E00      11770 AUTOFF LD      A,0      ;Check for auto XOFF
346C B7        11780      OR       A          ;On?
346D C8        11790      RET       Z          ;Quit if not
346E 91        11800      SUB      C          ;Matched char?
346F C0        11810      RET       NZ        ;Quit if not
3470 324534    11820      LD      (TASK8B+1),A ;Pause xmit (XOFF)
3473 C9        11830      RET
           11840 ;
3474           11850 TASKK @@KBD          ;Scan the keyboard
3474 3E08      00114      LD      A,8
3476 EF        00115      RST      40
3477 C0        11860      RET       NZ        ;Error (or no key depressed)
3478 FE80      11870      CP       BREAK     ;Ck for brk 1st
347A 2815      11880      JR       Z,ISBRK   ;Go on a Break
347C B7        11890      OR       A          ;Then for high bit set
347D FA6831    11900      JP       M,CMDKEY  ;Go if FN key
3480 06FF      11910 KISW  LD      B,0FFH    ;KI On/Off
3482 04        11920      INC      B
3483 05        11930      DEC      B
3484 C8        11940      RET       Z          ;Ret if KI is off
3485 DDE5      11950 NOTBRK PUSH     IX
3487 DD219538  11960      LD      IX,KIVCTR   ; else put key into
348B CDAE38    11970      CALL     OUTPGM    ; the output buffer
348E DDE1      11980      POP      IX
3490 C9        11990      RET
           12000 ;
           12010 ISBRK
3491 F3        12020      DI
3492 11E437    12030      LD      DE,CLDCB   ;Pt to *CL
3495 0E01      12040      LD      C,1      ;Send CTL 1, a
3497           12050      @@CTL          ; Break request
3497 3E05      00116      LD      A,5
3499 EF        00117      RST      40
349A FB        12060      EI
349B 01A138    12070      LD      BC,CLSEND
349E CD5533    12080      CALL     FILR1     ;Reset the CL buffer
34A1 CDDA33    12090      CALL     FS_OFF   ;Turn off the *FS
34A4 010020    12100      LD      BC,2000H   ;Time delay
34A7           12110      @@PAUSE
34A7 3E10      00118      LD      A,16
34A9 EF        00119      RST      40
34AA           12120      @@CKBRKC        ;Reset the break bit
34AA 3E6A      00120      LD      A,106
34AC EF        00121      RST      40

```

Program Code Section

```

34AD 0E00      12130      LD      C,0          ;Init the character
34AF 11E437    12140      LD      DE,CLDCB      ;P/u the CL DCB
34B2 F3        12150      DI
34B3           12160      @@PUT                ;Send the 0
34B3 3E04      00122      LD      A,4
34B5 EF        00123      RST      40
34B6 FB        12170      EI
34B7 C9        12180      RET
                12190      ;
                12200      ;      INTERRUPT TASK 9
                12210      ;      Only if RS232 does not interrupt
                12220      ;
                12230      IF      .NOT.BUFFRD
                12240 TCB9    DW      TASK9          ;Task ept
                12250      ENDIF
34B8 0603      12260 TASK9 LD      B,3          ;Max chars/pass
34BA 0E00      12270 PRL0OP LD      C,0          ;Test printer status
34BC 110000    12280      LD      DE,$-$          ;PDCB$
34BD           12290 PRDCB EQU      $-2
34BF           12300      @@CTL                ;Check the status
34BF 3E05      00124      LD      A,5
34C1 EF        00125      RST      40
34C2 C0        12310      RET      NZ          ;Ret if unavailable
34C3 DDE5      12320      PUSH     IX
34C5 DD219938  12330      LD      IX,PRVCTR        ;Get char from printer
                12340      IF      BUFFRD
34C9 CDB438    12350      CALL     PGMGET
                12360      ELSE
                12370      CALL     BUFGET          ;Buffer if available
                12380      ENDIF
34CC DDE1      12390      POP      IX
34CE C8        12400      RET      Z          ;None to get, back
34CF 4F        12410      LD      C,A
34D0           12420      @@PRT                ;Output to printer
34D0 3E06      00126      LD      A,6
34D2 EF        00127      RST      40
34D3 10E5      12430      DJNZ     PRL0OP        ;Loop if more
34D5 C9        12440      RET
                12450      ;
                12460      ;      Common routine to stuff various buffers
                12470      ;
34D6 DD6E00    12480 OUTPUT LD      L,(IX)        ;P/u pointer to
34D9 DD6601    12490      LD      H,(IX+1)        ; buffer PUT
34DC 77        12500      LD      (HL),A          ;Write char into buffer
34DD DD3400    12510      INC      (IX)          ;Bump buffer pointer
34E0 C0        12520      RET      NZ          ;Go if still in same page
34E1 CD3635    12530      CALL     NEXTAP         ;Find next avail page
34E4 2810      12540      JR      Z,DUMPCHR      ;Go if no pages available
34E6 DD7701    12550      LD      (IX+1),A        ;Set index to new page
34E9 21AD38    12560      LD      HL,FREEPG       ;Reduce the amount of
34EC 35        12570      DEC      (HL)          ; free pages
34ED 3E07      12580      LD      A,7          ;Less than 2K available?
34EF BE        12590      CP      (HL)
34F0 D8        12600      RET      C          ; & return with NZ
34F1 322430    12610      LD      (MAINLP+1),A    ;Set flag for warning
34F4 B7        12620      OR      A          ;Ensure NZ return
34F5 C9        12630      RET
                12640      ;
                12650      ;      No more pages available - keep last page

```

Program Code Section

```

12660 ;
34F6 DD3500 12670 DUMPCHR DEC (IX) ;Dump character and
34F9 AF 12680 XOR A ; return
34FA C9 12690 RET
12700 ;
12710 ; The following code is not executed, as it is too
12720 ; slow at rates >= 1200 baud because interupts are on.
12730 ; DE must be loaded with KIVCTR.
12740 ;
34FB 00 12750 DB 0
34FC DDE5 12760 PUSH IX ;Dev requesting the output
34FE E1 12770 POP HL
34FF AF 12780 XOR A ;The difference will be
3500 ED52 12790 SBC HL,DE ; the offset into the
3502 116A3E 12800 LD DE,DEVICE$ ; name table
3505 19 12810 ADD HL,DE
3506 010400 12820 LD BC,4
3509 11893E 12830 LD DE,OVRRUN$+3
350C EDB0 12840 LDIR
350E 21863E 12850 LD HL,OVRRUN$ ;Display the buffer
3511 12860 @@DSPLY ; overrun error
00128 IFEQ 00H,1
00129 LD HL,
00130 ENDIF
3511 3E0A 00131 LD A,10
3513 EF 00132 RST 40
3514 AF 12870 XOR A ; reuse current page
3515 C9 12880 RET
12890 ;
12900 ; Check for character available in dynamic buffer
12910 ;
3516 DD6E02 12920 BUFGET LD L,(IX+2) ;P/u pointer to next
3519 DD6603 12930 LD H,(IX+3) ; buffer GET
351C 7D 12940 LD A,L ;Check on in=out lo-order
351D DDBE00 12950 CP (IX)
3520 2005 12960 JR NZ,INNEOUT ;Go if in not equal out
3522 7C 12970 LD A,H ;Check on in=out hi-order
3523 DDBE01 12980 CP (IX+1)
3526 C8 12990 RET Z ;Ret if none to i/o
13000 ;
13010 ; Buffer is not empty - Get next character
13020 ;
3527 7E 13030 INNEOUT LD A,(HL) ;Get a char from buffer
3528 DD3402 13040 INC (IX+2) ;Advance lo-order pointer
352B C0 13050 RET NZ ;Ret if still same page
352C F5 13060 PUSH AF ;Save the character
352D CD4E35 13070 CALL FNPIU ;Find next page in use
3530 DD7703 13080 LD (IX+3),A ;Stuff new page index
3533 F1 13090 POP AF ;Recover the character
3534 25 13100 DEC H ;Set NZ return for rcvd
3535 C9 13110 RET
13120 ;
13130 ; Routine to find next available page buffer
13140 ;
3536 6C 13150 NEXTAP LD L,H ;Point to page buffer
3537 2639 13160 LD H,LINKS<-8 ; index
3539 3A0039 13170 LD A,(LINKS) ;Get next empty link
353C E5 13180 PUSH HL ;Save this index pointer
353D 6F 13190 LD L,A ;Point to new link

```

Program Code Section

```

353E 7E      13200      LD      A,(HL)      ;Get what it links to
353F B7      13210      OR       A          ;Test if none left
3540 2002     13220      JR       NZ,GOTNAP   ;Go if still more
3542 E1      13230      POP      HL         ;Restore reg & return
3543 C9      13240      RET              ; with Z-flag for error
          13250 ;
          13260 ;      Found the next available page - set the links
          13270 ;
3544 320039   13280 GOTNAP LD      (LINKS),A    ;Update new next avail
3547 7D      13290      LD      A,L        ;Xfer index of new page
3548 E1      13300      POP      HL         ;Rcvr pointer to index
3549 77      13310      LD      (HL),A      ; of old & link to new
354A 6F      13320      LD      L,A        ;Repoint to new page's
354B 3600     13330      LD      (HL),0     ; index & show it is
354D C9      13340      RET              ; the last one
          13350 ;
          13360 ;      Find next page in use
          13370 ;
354E 3AAD38   13380 FNPIU LD      A,(FREEPG)  ;Show one additional
3551 3C      13390      INC      A          ; page is free
3552 32AD38   13400      LD      (FREEPG),A
3555 3A0139   13410      LD      A,(HIPAGE)   ;P/u highest page avail
3558 6F      13420      LD      L,A        ;Set HL to its index
3559 7C      13430      LD      A,H
355A 2639     13440      LD      H,LINKS<-8   ;Show that page links to
355C 77      13450      LD      (HL),A      ; the one we just emptied
355D 320139   13460      LD      (HIPAGE),A   ;Now update the new end
3560 6F      13470      LD      L,A        ;Set HL to the emptied
3561 7E      13480      LD      A,(HL)      ; page, p/u what it
3562 3600     13490      LD      (HL),0     ; linked to, & show old
3564 C9      13500      RET              ; is end. Ret A=link
          13510 ;
          13520 ;      Execute a DOS command
          13530 ;
3565 21FF2F   13540 DOSCMD LD      HL,BASE-1
3568 0601     13550      LD      B,1          ;Set LOW$
356A         13560      @@HIGH$
356A 3E64     00133      LD      A,100
356C EF      00134      RST      40
356D 219A35   13570      LD      HL,CMDPMT    ;Issue prompt
3570         13580      @@DSPLY
          00135      IFEQ 00H,1
          00136      LD      HL,
          00137      ENDIF
3570 3E0A     00138      LD      A,10
3572 EF      00139      RST      40
3573 010050   13590      LD      BC,80<8      ;Max characters
3576 214438   13600      LD      HL,DUMMY     ;=>input buffer
3579         13610      @@KEYIN              ;Get command request
3579 3E09     00140      LD      A,9
357B EF      00141      RST      40
357C D8      13620      RET      C          ;Back on Break
357D 04      13630      INC      B
357E 05      13640      DEC      B
357F C8      13650      RET      Z          ; or CR only
3580 EB      13660      EX       DE,HL
3581 210000   13670      LD      HL,$-$      ;Pt to CFLAG$
3582         13680 CFLAG EQU      $-2
3584 CB46     13690      BIT      0,(HL)     ;Get current status

```

Program Code Section

3586	E5	13700	PUSH	HL		
3587	F5	13710	PUSH	HL		
3588	CBC6	13720	SET	0,(HL)		;Save memory freeze status
358A	EB	13730	EX	DE,HL		;Freeze memory
358B		13740	@@CMNDR			
358B	3E19	00142	LD	A,25		;Do the command
358D	EF	00143	RST	40		
358E	21AB35	13750	LD	HL,CMLPTD		;Show cmd finished
3591		13760	@@DSPLY			
		00144	IFEQ	00H,1		
		00145	LD	HL,		
		00146	ENDIF			
3591	3E0A	00147	LD	A,10		
3593	EF	00148	RST	40		
3594	F1	13770	POP	AF		;Get the previous status
3595	E1	13780	POP	HL		; and CFLAG\$ location
3596	C0	13790	RET	NZ		;Back if was set before
3597	CB86	13800	RES	0,(HL)		; else restore it
3599	C9	13810	RET			
359A	0A	13820	CMDPMT	DB	LF,LF,'Enter command:',CR	
	0A 45 6E 74 65 72 20 63					
	6F 6D 6D 61 6E 64 3A 0D					
35AB	0A	13830	CMLPTD	DB	LF,'Command completed',CR	
	43 6F 6D 6D 61 6E 64 20					
	63 6F 6D 70 6C 65 74 65					
	64 0D					
		13840 ;				
		13850 ;				
		13860 ;	Messages			
35BE	7B	13870	BRACKET	DB	'{ }',3	;Brackets around hex byte
	20 20 7D 03					
35C3	1D	13880	FILEPMT	DB	29,10,'File name: ',3	
	0A 46 69 6C 65 20 6E 61					
	6D 65 3A 20 03					
35D1	1D	13890	OPENMSG	DB	29,10,'File already open',CR	
	0A 46 69 6C 65 20 61 6C					
	72 65 61 64 79 20 6F 70					
	65 6E 0D					
35E5	0A	13900	MNUMSG	DB	LF	
35E6	20	13910	STAT1	DB	'	
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 20					
360A	20	13920		DB	'	,LF
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 20 20 20 20 20 20					
	20 20 0A					
362E	44	13930		DB	'DUPLX ECHO ECOLF ACCLF REWND PEOF '	
	55 50 4C 58 20 45 43 48					
	4F 20 20 45 43 4F 4C 46					
	20 41 43 43 4C 46 20 52					
	45 57 4E 44 20 50 45 4F					
	46 20 20					
3652	20	13940		DB	' DCC CLS 8-B CMD HND SH EXIT',LF	
	44 43 43 20 20 20 43 4C					

Program Code Section

```

    53 20 20 20 38 2D 42 20
    20 20 43 4D 44 20 20 48
    4E 44 53 48 20 20 45 58
    49 54 0A
3676 3D      13950      DB      '==1== ==2== ==3== ==4== ==5== ==6== '
    3D 31 3D 3D 20 3D 3D 32
    3D 3D 20 3D 3D 33 3D 3D
    20 3D 3D 34 3D 3D 20 3D
    3D 35 3D 3D 20 3D 3D 36
    3D 3D 20
369A 3D      13960      DB      '==7== ==8== ==9== ==0== '
    3D 37 3D 3D 20 3D 3D 38
    3D 3D 20 3D 3D 39 3D 3D
    20 3D 3D 30 3D 3D 20
    13970      IF
36B2 3D      13980      DB      '@MOD4
    3D 3A 3D 3D 20 3D 3D 2D      '==:== ====='
    3D 3D
    13990      ENDIF
    14000      IF      @MOD2
    14010      DB      '==--== ==--=='
    14020      ENDIF
36BD 0A      14030      DB      LF
36BE 20      14040      DB      ' *KI      *DO      *PR      *CL      *FS      *FR '
    2A 4B 49 20 20 20 2A 44
    4F 20 20 20 2A 50 52 20
    20 20 2A 43 4C 20 20 20
    2A 46 53 20 20 20 2A 46
    52 20 20
36E2 20      14050      DB      ' DTD      ???      ID      RES      ON      OFF ',LF
    44 54 44 20 20 20 3F 3F
    3F 20 20 20 49 44 20 20
    20 20 52 45 53 20 20 20
    4F 4E 20 20 20 20 4F 46
    46 0A
3705 20      14060      STAT2  DB      '
    20 20 20 20 20 20 20 20
    20 20 20 20 20 20 20 20
    20 20 20 20 20 20 20 20
    20 20 20 20 20 20 20 20
    20 20 20 20
372A 20      14070      DB      ' ',CR
    20 0D
372D 8134      14080      STATAB  DW      KISW+1,STAT2+2,DEVOUT+1,STAT2+8
    0737 1731 0D37
3735 3B31      14090      DW      PUTPR+1,STAT2+14,TASK8B+1,STAT2+19
    1337 4534 1837
373D FA33      14100      DW      TASK8A+2,STAT2+21,FSSW+1,STAT2+26
    1A37 4130 1F37
3745 2E31      14110      DW      FRSW+1,STAT2+32,FRIOSW+1,STAT2+38
    2537 4831 2B37
374D 6A30      14120      DW      DPLXSW+1,STAT1+2,ECHOSW+1,STAT1+8
    E835 C930 EE35
3755 0A31      14130      DW      ECOLF,STAT1+14,ACCLFSW+1,STAT1+20
    F435 DB30 FA35
375D 8930      14140      DW      DSPCTRL+1,STAT1+38,EIGHT+1,STAT1+50
    0C36 1F34 1836
3765 A330      14150      DW      SHAKE+1,STAT1+61
    2336

```

Program Code Section

```

3769 46      14160 FSNAME$ DB      'FS-Spec: ',3
      53 2D 53 70 65 63 3A 20
      03
3773 20      14170 FRNAME$ DB      '  FR-Spec: ',3
      20 46 52 2D 53 70 65 63
      3A 20 03
377F 20      14180 PAGSPR$ DB      '  Memory:  K',CR
      20 4D 65 6D 6F 72 79 3A
      20 20 20 4B 0D
378D 2A      14190 CMDERR$ DB      '** Invalid command sequence **',CR
      2A 20 49 6E 76 61 6C 69
      64 20 63 6F 6D 6D 61 6E
      64 20 73 65 71 75 65 6E
      63 65 20 2A 2A 0D
37AC 57      14200 LILPG$  DB      'Warning! Less than 2K of buffer left '
      61 72 6E 69 6E 67 21 20
      4C 65 73 73 20 74 68 61
      6E 20 32 4B 20 6F 66 20
      62 75 66 66 65 72 20 6C
      65 66 74 20
37D1 20      14210          DB      ' X-OFF transmitted',CR
      58 2D 4F 46 46 20 74 72
      61 6E 73 6D 69 74 74 65
      64 0D
      14220 ;
      14230 ;      File control blocks
      14240 ;
0020      14250 CLDCB  DS      32
0020      14260 FS_FCB DS      32
0020      14270 FR_FCB DS      32
0051      14280 DUMMY  DS      81      ;Used for dos cmd buffer also
      14290 ;
      14300 ;      Put/Get index pointers
      14310 ;
3895 0000      14320 KIVCTR DW      0,0
      0000
3899 0000      14330 PRVCTR DW      0,0
      0000
389D 0000      14340 CLREC  DW      0,0
      0000
38A1 0000      14350 CLSEND DW      0,0
      0000
38A5 0000      14360 FSVCTR DW      0,0
      0000
38A9 0000      14370 FRVCTR DW      0,0
      0000
0001      14380 FREEPG  DS      1
      14390 ;
      14400 ;      Routines  to buffer I/O in pgm loop
      14410 ;
38AE F3      14420 OUTPGM DI
38AF CDD634   14430 CALL      OUTPUT
38B2 FB      14440          EI
38B3 C9      14450          RET
38B4 F3      14460 PGMGET DI
38B5 CD1635   14470 CALL      BUFGET
38B8 FB      14480          EI
38B9 C9      14490          RET
      14500 ;

```


Program Code Section

```

      14510 ;      Page buffer Link table
      14520 ;
39000 14530      ORG      $<-8+1<+8
00001 14540 LINKS    DS      1      ;Link to next available
00001 14550 HIPAGE   DS      1      ;Link to last available
00001 14560          DS      1      ;Init to 1st avail
00001 14570          DS      1      ;Init to last avail
00FC  14580          DS      252     ;Space for linkage tables
      14590 ;
      14600 ;      Transmit and Receive File buffers
      14610 ;
01000 14620 XMTBUF    DS      256
01000 14630 RCVBUF    DS      256
      14640 ;
3C000 14650      SUBTTL  '<COMM initialization code>'
```

COMM initialization code

```

3C00      14670 *GET      LCOMMA:3          ;Initialization code
          14680 ;LCOMMA/ASM - COMM Initialization Code
          14690 ;
          14700 ;      Entry point to LCOMM
          14710 ;
          14720 LCOMM
3C00      14730 @@CKBRKC          ;Check for break
3C00 3E6A 00149 LD      A,106
3C02 EF 00150 RST      40
3C03 2804 14740 JR      Z,LCOMMA          ;Continue if not
3C05 21FFFF 14750 LD      HL,-1          ; else ABORT
3C08 C9 14760 RET
          14770 ;
3C09 F3 14780 LCOMMA DI
3C0A ED730430 14790 LD      (STACK),SP          ;Save for exit
3C0E E5 14800 PUSH     HL          ;Save ptr to CMD buffer
3C0F 210000 14810 LD      HL,0
3C12      14820 @@BREAK          ;Disable break vectoring
          00151 IFEQ     00H,1
          00152 LD      HL,
          00153 ENDIF
3C12 3E67 00154 LD      A,103
3C14 EF 00155 RST      40
3C15 FB 14830 EI
3C16 21533D 14840 LD      HL,HELLO$          ;Issue the copyright
3C19      14850 @@DSPLY
          00156 IFEQ     00H,1
          00157 LD      HL,
          00158 ENDIF
3C19 3E0A 00159 LD      A,10
3C1B EF 00160 RST      40
3C1C E1 14860 POP      HL
3C1D 11E437 14870 LD      DE,CLDCB          ;Point to FCB
3C20      14880 @@FSPEC          ;Get the *CL spec
3C20 3E4E 00161 LD      A,78
3C22 EF 00162 RST      40
3C23 C24A3D 14890 JP      NZ,BADCL          ;Go error if none
3C26 1A 14900 LD      A,(DE)
3C27 FE2A 14910 CP      '*'          ;Ck for device spec
3C29 C24A3D 14920 JP      NZ,BADCL          ;Go if not a device
3C2C 11A23E 14930 LD      DE,PRMTBL$
3C2F      14940 @@PARAM          ;Parse the parms
3C2F 3E11 00163 LD      A,17
3C31 EF 00164 RST      40
3C32 F5 14950 PUSH     AF          ;Save status
3C33 C41A30 14960 CALL    NZ,$ERROR          ;Display any error
3C36 F1 14970 POP      AF
3C37 C20A30 14980 JP      NZ,$ABORT          ; and then quit
          14990 ;
3C3A 0600 15000 LD      B,0
3C3C 11E437 15010 LD      DE,CLDCB          ;Open the comm line
3C3F      15020 @@OPEN
3C3F 3E3B 00165 LD      A,59
3C41 EF 00166 RST      40
3C42 F5 15030 PUSH     AF
3C43 C41A30 15040 CALL    NZ,$ERROR          ;Show any open error
3C46 F1 15050 POP      AF
3C47 C20A30 15060 JP      NZ,$ABORT          ; and then quit
3C4A 0E02 15070 LD      C,2          ;INIT function for hardware
3C4C      15080 @@CTL          ;Just in case

```

COMM initialization code

```

3C4C 3E05      00167      LD      A,5
3C4E EF        00168      RST      40
3C4F 21D63D    15090      LD      HL,GETMNU$      ;How the user gets menu
3C52           15100      @@DSPLY
           00169      IFEQ     00H,1
           00170      LD      HL,
           00171      ENDIF
3C52 3E0A      00172      LD      A,10
3C54 EF        00173      RST      40
3C55 AF        15110      XOR      A
3C56 320438    15120      LD      (FS_FCB),A      ;Init FCB's to OFF
3C59 322438    15130      LD      (FR_FCB),A
3C5C ED5BA03E  15140      LD      DE,(PRNAME)      ;Load 'PR' backwards
3C60           15150      @@GTDCB
3C60 3E52      00174      LD      A,82
3C62 EF        00175      RST      40
3C63 22BD34    15160      LD      (PRDCB),HL      ;Store address for @CTL
3C66           15170      @@FLAGS      ;Set up IY
3C66 3E65      00176      LD      A,101
3C68 EF        00177      RST      40
3C69 FDE5      15180      PUSH     IY
3C6B D1        15190      POP      DE
3C6C 211200    15200      LD      HL,'S'-'A'      ;Offset to SFLAG$
3C6F 19        15210      ADD      HL,DE
3C70 221130    15220      LD      (SFLG),HL      ;Store for later
3C73 210A00    15230      LD      HL,'K'-'A'      ;Offset to KFLAG$
3C76 19        15240      ADD      HL,DE
3C77 CB86      15250      RES      0,(HL)      ;Be sure BREAK bit is off
3C79 210200    15260      LD      HL,'C'-'A'      ;CFLAG$
3C7C 19        15270      ADD      HL,DE
3C7D 228235    15280      LD      (CFLAG),HL
3C80 CB4E      15290      BIT      1,(HL)      ;Doing CMNDR?
3C82 210000    15300      LD      HL,0
3C85 45        15310      LD      B,L
3C86 2801      15320      JR      Z,$+3      ;Use LOW$ if CMNDR
3C88 04        15330      INC      B
3C89           15340      @@HIGH$
3C89 3E64      00178      LD      A,100
3C8B EF        00179      RST      40
3C8C 23        15350      INC      HL      ;Available for use
3C8D 25        15360      DEC      H      ; by page buffers
3C8E 44        15370      LD      B,H      ;Set B to highest usable
3C8F 210039    15380      LD      HL,LINKS
3C92 3E3C      15390      LD      A,LCOMM<-8      ;Establish 1st usable
3C94 77        15400      LD      (HL),A      ;Init to 1st available
3C95 2C        15410      INC      L      ; page buffer
3C96 70        15420      LD      (HL),B      ;Init to highest page
3C97 2C        15430      INC      L      ; buffer available
3C98 77        15440      LD      (HL),A      ;Init to begin & highest
3C99 2C        15450      INC      L
3C9A 70        15460      LD      (HL),B
           15470 ;
           15480 ;      Establish page buffer linkage table
           15490 ;
3C9B 6F        15500 DOLINKS LD      L,A      ;Init memory begin to
3C9C 3C        15510      INC      A      ; high bytes for as many
3C9D 77        15520      LD      (HL),A      ; bytes as pages to top
3C9E B8        15530      CP      B
3C9F 20FA      15540      JR      NZ,DOLINKS

```

COMM initialization code

```

3CA1 6F      15550      LD      L,A
3CA2 3600     15560      LD      (HL),0      ;Close out with zero
                15570 ;
                15580 ;      Establish starting page buffers for devices
                15590 ;
3CA4 2604     15600      LD      H,4      ;Init 1st at links+4
3CA6 DD219538 15610      LD      IX,KIVCTR
3CAA CDE63C   15620      CALL    INITBUF      ;Init *KI page buffer
3CAD DD219938 15630      LD      IX,PRVCTR
3CB1 CDE63C   15640      CALL    INITBUF      ;Init *PR page buffer
3CB4 DD219D38 15650      LD      IX,CLREC
3CB8 CDE63C   15660      CALL    INITBUF      ;Init *CL-R page buffer
3CBB DD21A138 15670      LD      IX,CLSEND
3CBF CDE63C   15680      CALL    INITBUF      ;Init *CL-S page buffer
3CC2 DD21A538 15690      LD      IX,FSVCTR
3CC6 CDE63C   15700      CALL    INITBUF      ;Init *FS page buffer
3CC9 DD21A938 15710      LD      IX,FRVCTR
3CCD CDE63C   15720      CALL    INITBUF      ;Init *FR page buffer
                15730 ;
                15740 ;      Calculate free buffer space
                15750 ;
3CD0 2639     15760      LD      H,LINKS<-8      ;P/u hi-order link table
3CD2 0600     15770      LD      B,0      ;Init count to zero
3CD4 3A0039   15780      LD      A,(LINKS)      ;Find pointer to 1st spr
3CD7 6F      15790      LD      L,A
3CD8 7E      15800 FBS1   LD      A,(HL)      ;P/u pointer to next
3CD9 B7      15810      OR      A      ; spare & test if last
3CDA 2804     15820      JR      Z,FBS2      ;Exit if no more
3CDC 04      15830      INC     B      ;Bump counter
3CDD 6F      15840      LD      L,A      ;Show new pointer
3CDE 18F8     15850      JR      FBS1
3CE0 78      15860 FBS2   LD      A,B      ;Transfer the count
3CE1 32AD38   15870      LD      (FREEPG),A      ; and save it
3CE4 1818     15880      JR      SETUPT
                15890 ;
                15900 ;      Routine to establish starting page buffers
                15910 ;
3CE6 DD360000 15920 INITBUF LD      (IX),0      ;Show low-order PUT/GET
3CEA DD360200 15930      LD      (IX+2),0      ; start at 0 reference
3CEE E5      15940      PUSH   HL
3CEF CD3635   15950      CALL    NEXTAP      ;Find next available page
3CF2 CA463D   15960      JP      Z,NOBUFS      ;Go if insufficient pages
3CF5 E1      15970      POP     HL
3CF6 DD7701   15980      LD      (IX+1),A      ;Set high-order PUT/GET
3CF9 DD7703   15990      LD      (IX+3),A      ; page index pointers
3CFC 24      16000      INC     H      ;Bump to next entry in
3CFD C9      16010      RET      ; link table & return
                16020 ;
                16030 ;      Routine to set up the task processor
                16040 ;
                16050 SETUPT
                16060
                16070      IF      .NOT.BUFFRD
                16080      LD      DE,TCB8      ;CL task process
                16090      LD      C,8
3CFE          16090      @@ADTSK
                16100      LD      A,29
                16110      RST     40
                16120      LD      DE,TCB9      ;Printer output task
                16130      LD      C,9      ;Only if RS232 does

```

COMM initialization code

```

3CFE      16120      @@ADTSK      ; not interrupt
          00182      LD      A,29
          00183      RST     40
          16130      ENDIF
          16140      ;
          16150      IF      BUFFERD
3CFE 11E437 16160      LD      DE,CLDCB      ;Turn on wakeup feature
3D01 FD21F833 16170      LD      IY,TASK8A      ;Wakeup driver address
3D05 0E04 16180      LD      C,4      ;Set addr CTL value
3D07 F3 16190      DI
3D08      16200      @@CTL      ;Send to Com driver
3D08 3E05 00184      LD      A,5
3D0A EF 00185      RST     40
3D0B FB 16210      EI
3D0C FD221432 16220      LD      (OLDVEC),IY      ;Save previous state
          16230      ENDIF
          16240      ;
3D10 21EE3D 16250      LD      HL,LFEEDS      ;Clear most of screen
3D13      16260      @@DSPLY
          00186      IFEQ     00H,1
          00187      LD      HL,
          00188      ENDIF
3D13 3E0A 00189      LD      A,10
3D15 EF 00190      RST     40
          16270      ;
          16280      ;      Transfer any translation characters
          16290      ;
3D16 3AEA3E 16300      LD      A,(XLATES+1)      ;Transfer the output
3D19 326430 16310      LD      (XLTS1+1),A      ; translation character
3D1C 3AE93E 16320      LD      A,(XLATES)
3D1F 326830 16330      LD      (XLTS2+1),A
          16340      ;
3D22 3AEC3E 16350      LD      A,(XLATER+1)      ;Transfer the input
3D25 322B34 16360      LD      (XLTR1+1),A      ; translation character
3D28 3AEB3E 16370      LD      A,(XLATER)
3D2B 322F34 16380      LD      (XLTR2+1),A
          16390      ;
3D2E 3AE33E 16400      LD      A,(NULLPRM)      ;Transfer the null parm
3D31 323434 16410      LD      (ACCNUL+1),A
3D34 3AE53E 16420      LD      A,(XONP)      ;Transfer the XON/XOFF
3D37 32A930 16430      LD      (XONP1),A      ; parms
3D3A 3AE73E 16440      LD      A,(XOFFP)
3D3D 32AD30 16450      LD      (XOFFP1),A
3D40 322F30 16460      LD      (XOFFP2),A
3D43 C32330 16470      JP      MAINLP
          16480      ;
          16490      ;      Error handling on initialization
          16500      ;
3D46 21413E 16510      NOBUFS LD      HL,NOBUFS$      ;"Not enuf mem for buffers
3D49 DD 16520      DB      0DDH
3D4A 21223E 16530      BADCL LD      HL,BADCL$      ;"Need RS-232 device name
3D4D      16540      @@LOGOT
          00191      IFEQ     00H,1
          00192      LD      HL,
          00193      ENDIF
3D4D 3E0C 00194      LD      A,12
3D4F EF 00195      RST     40
3D50 C30A30 16550      JP      $ABORT
          16560      ;

```

COMM initialization code

```

                16570 ;      Messages
                16580 ;
3D53 43        16590 HELLO$ DB      'COMM'
          4F 4D 4D
3D57          16600 *GET      CLIENT:3
                16610 ;CLIENTS/ASM - File to establish sign-on headers
                16620 ;
3D57 20        16630          DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
          2D 20 36 2E 32 2E 30 20
          2D 20 43 6F 70 79 72 69
          67 68 74 20 31 39 38 32
          2F 38 33 2F 38 34 20 62
          79 20 4C 6F 67 69 63 61
          6C
3D81 20        16640          DB      ' Systems, Inc.      ',10
          53 79 73 74 65 6D 73 2C
          20 49 6E 63 2E 20 20 20
          20 20 20 0A
                16650 ;
3D96 41        16660          DB      'All Rights Reserved. Licensed 1982/83/84'
          6C 6C 20 52 69 67 68 74
          73 20 52 65 73 65 72 76
          65 64 2E 20 4C 69 63 65
          6E 73 65 64 20 31 39 38
          32 2F 38 33 2F 38 34
3DBE 20        16670          DB      ' to xxxxxxxxxxxxxxxxxxx',10,13
          74 6F 20 78 78 78 78 78
          78 78 78 78 78 78 78 78
          78 78 78 78 78 0A 0D
                16680 IF      @MOD4
3DD6 55        16690 GETMNU$ DB      'Use <CLEAR-8> for menu',LF,CR
          73 65 20 3C 43 4C 45 41
          52 2D 38 3E 20 66 6F 72
          20 6D 65 6E 75 0A 0D
                16700 ENDIF
                16710 IF      @MOD2
                16720 GETMNU$ DB      'Use <ESC-8> for menu',LF,CR
                16730 ENDIF
3DEE 0A        16740 LFEEDS DB      LF,LF,LF,LF,LF,LF,LF
          0A 0A 0A 0A 0A 0A
3DF 5 0A        16750          DB      LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,14,3
          0A 0A 0A 0A 0A 0A 0A 0A
          0A 0A 0E 03
                16760 ;
3E02 00        16770          DC      32,0      ;Patch space
          00 00 00 00 00 00 00 00
          00 00 00 00 00 00 00 00
          00 00 00 00 00 00 00 00
          00 00 00 00 00 00
                16780 ;
3E22 43        16790 BADCL$ DB      'Comm Line driver not specified',CR
          6F 6D 6D 20 4C 69 6E 65
          20 64 72 69 76 65 72 20
          6E 6F 74 20 73 70 65 63
          69 66 69 65 64 0D
3E41 49        16800 NOBUFS$ DB      'Insufficient memory to establish buffers',CR
          6E 73 75 66 66 69 63 69
          65 6E 74 20 6D 65 6D 6F
          72 79 20 74 6F 20 65 73
          74 61 62 6C 69 73 68 20

```

COMM initialization code

```

        62 75 66 66 65 72 73 0D
3E6A 20      16810 DEVICE$ DB      ' KI  PR CL-RCL-S FS  FR ????'
        4B 49 20 20 50 52 20 43
        4C 2D 52 43 4C 2D 53 20
        46 53 20 20 46 52 20 3F
        3F 3F 3F
3E86 2A      16820 OVRUN$ DB      '** xxxx Buffer overrun **',3
        2A 20 78 78 78 78 20 42
        75 66 66 65 72 20 6F 76
        65 72 72 75 6E 20 2A 2A
        03
3EA0 50      16830 PRNAME  DB      'PR'
        52
        16840 ;
3EA2 58      16850 PRMTBL$ DB      'XLATES'
        4C 41 54 45 53
3EA8 E93E    16860          DW      XLATES
3EAA 58      16870          DB      'XS  '
        53 20 20 20 20
3EB0 E93E    16880          DW      XLATES
3EB2 58      16890          DB      'XLATER'
        4C 41 54 45 52
3EB8 EB3E    16900          DW      XLATER
3EBA 58      16910          DB      'XR  '
        52 20 20 20 20
3EC0 EB3E    16920          DW      XLATER
3EC2 4E      16930          DB      'NULL '
        55 4C 4C 20 20
3EC8 E33E    16940          DW      NULLPRM
3ECA 4E      16950          DB      'N  '
        20 20 20 20 20
3ED0 E33E    16960          DW      NULLPRM
3ED2 58      16970          DB      'XON  '
        4F 4E 20 20 20
3ED8 E53E    16980          DW      XONP
3EDA 58      16990          DB      'XOFF '
        4F 46 46 20 20
3EE0 E73E    17000          DW      XOFFP
3EE2 00      17010          NOP
        17020 ;
3EE3 FFFF    17030 NULLPRM DW      -1          ;Default to accept nulls
3EE5 1100    17040 XONP   DW      'Q'-40H      ;Ctl-Q
3EE7 1300    17050 XOFFP  DW      'S'-40H      ;Ctl-S
3EE9 0000    17060 XLATES DW      0
3EEB 0000    17070 XLATER DW      0
        17080 ;
3EED        00110          SUBTTL <>
3C00        00120          END      LCOMM

```

\$ABORT	300A \$ERROR	301A \$EXIT	3000
\$OPEN	300F @@1	0000 @@2	0000
@@3	0000 @@4	0000 @MOD2	0000
@MOD4	FFFF ACCLFSW	30DA ACCNUL	3433
AUTOFF	346A BADCL	3D4A BADCL\$	3E22
BASE	3000 BRACKET	35BE BREAK	0080
BUFFRD	FFFF BUFGET	3516 CFLAG	3582
CKFREPG	3045 CLDCB	37E4 CLOUT	30FA
CLREC	389D CLS	328F CLSEND	38A1
CMDERR	327D CMDERR\$	378D CMDKEY	3168
CMDPMT	359A CMPLTD	35AB CR	000D
CRSW	30E0 CTLQ	30B2 CTRLR	30C2
CVD1	3330 CVD2	3335 DEVICE\$	3E6A
DEVOUT	3116 DOLINKS	3C9B DOSCMD	3565
DPLXSW	3069 DSPCTRL	3088 DUMMY	3844
DUMPCHR	34F6 ECHOSW	30C8 ECLF1	3109
ECOLF	310A EIGHT	341E ENUFG	3037
EOFFS	305C EXIT	320F FBS1	3CD8
FBS2	3CE0 FILEOF	338D FILEPMT	35C3
FILFR	33CC FILID	3396 FILR1	3355
FILR2	3360 FILR3	3369 FILR4	336B
FILRES	334C FILREW	3384 FNDDVR	3400
FNPIU	354E FRCIT	3467 FRCPUT	3451
FREEPG	38AD FRIOSW	3147 FRI0 OFF	33E4
FRNAME\$	3773 FRSW	312D FRVCTR	38A9
FR FCB	3824 FR OFF	33DF FSNAME\$	3769
FSOFF	3077 FSSW	3040 FSSWGO	3050
FSVCTR	38A5 FS FCB	3804 FS OFF	33DA
GETMNU\$	3DD6 GOTNAP	3544 HELLO\$	3D53
HIPAGE	3901 INITBUF	3CE6 INNEOUT	3527
ISBRK	3491 KEEPCH	3438 KISW	3480
KIVCTR	3895 LCMON	3070 LCOMM	3C00
LCOMMA	3C09 LF	000A LFEEDS	3DEE
LILPG\$	37AC LINKS	3900 LNKD	341A
MAINLP	3023 MENU	329A MNUMSG	35E5
NEXTAP	3536 NOBUFS	3D46 NOBUFS\$	3E41
NOSQ	30B8 NOTBRK	3485 NOTCLS	3129
NOTCR	30DF NOTNOW	33D3 NULLPRM	3EE3
OFFS	337C OLDVEC	3214 OPENMSG	35D1
OUTPGM	38AE OUTPUT	34D6 OVRUN\$	3E86
PAGSPR\$	377F PGMGET	38B4 POPERR	3267
PRDCB	34BD PRLOOP	34BA PRMTBL\$	3EA2
PRNAME	3EA0 PRVCTR	3899 PUTPR	313A
QCL	3238 QFUNC	3233 QONOFF	325A
QONOFF1	325F QSHAKE	3242 QSHAKE1	3251
QUIT\$	3003 RCVBUF	3B00 RETADD	341D
SAVCHR	30A1 SENDIT	3062 SETUPT	3CFE
SFLG	3011 SHAKE	30A2 SKIPREC	30F4
SNDOUT	3102 STACK	3004 STAT1	35E6
STAT2	3705 STATAB	372D STATLP1	32B4
STATLP2	32D8 STATLP3	3301 STATLP4	3324
TAKEREC	30F1 TASK8A	33F8 TASK8B	3444
TASK9	34B8 TASKK	3474 TASKS	33E9
TSTNUL	3430 TURNOF	3284 TURNON	3286
XLATER	3EEB XLATES	3EE9 XLTR1	342A
XLTR2	342E XLTS1	3063 XLTS2	3067
XMTBUF	3A00 XOFF	0013 XOFFP	3EE7
XOFFP1	30AD XOFFP2	302F XONP	3EE5

XONP1	30A9 @@ABORT	6C16 @@ADTSK	6CA9
@@BANK	71C1 @@BKSP	6EA1 @@BREAK	71D7
@@CHNIO	6C01 @@CKBRKC	7225 @@CKDRV	6CFD
@@CKEOF	6EB6 @@CKTSK	6C94 @@CLOSE	6E8C
@@CLS	720F @@CMNDI	6C40 @@CMNDR	6C55
@@CTL	6A65 @@DATE	6BD7 @@DCSTAT	6D3C
@@DEBUG	6C7F @@DECHEX	7141 @@DIRRD	70AE
@@DIRWR	70C3 @@DIV16	712C @@DIV8	7117
@@DODIR	6D12 @@DSP	6A29 @@DSPLY	6AC9
@@ERROR	6C6A @@EXIT	6C2B @@FEXT	701B
@@FLAGS	71AB @@FNAME	7030 @@FSPEC	7006
@@GATRD	7099 @@GATWR	70D8 @@GET	6A3D
@@GTDCB	705A @@GTDCB	7045 @@GTMOD	706F
@@HDFMT	6DE4 @@HEX16	7180 @@HEX8	716B
@@HEXDEC	7156 @@HIGH\$	7195 @@INIT	6E62
@@KBD	6AA1 @@KEY	6A15 @@KEYIN	6AB5
@@KLTSK	6CE8 @@LOAD	6FDC @@LOC	6ECB
@@LOF	6EE0 @@LOGGER	6B00 @@LOGOT	6B15
@@MSG	6B4C @@MUL16	7102 @@MUL8	70ED
@@OPEN	6E77 @@PARAM	6BC2 @@PAUSE	6BAD
@@PEOF	6EF5 @@POSN	6F0A @@PRINT	6B61
@@PRT	6A79 @@PUT	6A51 @@RAMDIR	6D27
@@RDSEC	6DBA @@RDSSC	7084 @@READ	6F1F
@@REMOV	6E4D @@RENAM	6E38 @@REW	6F34
@@RMTSK	6CBE @@RPTSK	6CD3 @@RREAD	6F49
@@RSLCT	6DA5 @@RSTOR	6D66 @@RUN	6FF1
@@RWRTIT	6F5E @@SEEK	6D90 @@SEEKSC	6F73
@@SKIP	6F88 @@SLCT	6D51 @@STEPI	6D7B
@@TIME	6BEC @@VDCTL	6B98 @@VER	6F9D
@@VRSEC	6DCF @@WEOF	6FB2 @@WHERE	6A8D
@@WRITE	6FC7 @@WRSEC	6DF9 @@WRSSC	6E0E
@@WRTRK	6E23		
00000 Total errors			

NOTES:

NOTES:

CONV/CMD - Convert Model III TRSDOS files

The Conv utility will move files from TRSDOS 1.2 and 1.3, Model III, copying them to an LS-DOS or TRSDOS 6 disk.

```

00100 ;CONV/ASM - Convert TRSDOS 1.2, 1.3 Disks
0000 00110 TITLE <CONV - LS-DOS 6.2>
00120 ;
001C 00130 HOME EQU 1CH
001F 00140 CLR EQU 1FH
0003 00150 ETX EQU 03H
000D 00160 CR EQU 0DH
000A 00170 LF EQU 10
00180 ;
0040 00190 FLAG EQU 01000000B
0010 00200 ABB EQU 00010000B
00210 ;
0000 00220 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00230 *GET COPYCOM:3 ;Copyright message
03920 ; COPYCOM - File for Copyright COMMENT block
03930 ;
0000 03940 COM '<*(C) 1982,83,84 by LSI*>'
00240 ;
2600 00250 ORG 2600H
00260 ;
00270 BEGIN
2600 00280 @@CKBRKC
2600 3E6A 00001 LD A,106
2602 EF 00002 RST 40
2603 2804 00290 JR Z,BEGINA ;Continue if no break
2605 21FFFF 00300 LD HL,-1
2608 C9 00310 RET ; else abort
00320 ;
00330 BEGINA
2609 ED734626 00340 LD (STACK),SP ;Save entry stack
260D E5 00350 PUSH HL ;Save ptr to CMD buffer
260E 00360 @@DSPLY HELLO$ ;Display the signon
00003 IFEQ 01H,1
260E 21C32A 00004 LD HL,HELLO$
00005 ENDIF
2611 3E0A 00006 LD A,10
2613 EF 00007 RST 40
2614 210000 00370 LD HL,0 ;Set up to get HIGH$
2617 45 00380 LD B,L
2618 00390 @@FLAGS ;IY => flag table base
2618 3E65 00008 LD A,101
261A EF 00009 RST 40
261B FDCB024E 00400 BIT 1,(IY+'C'-'A') ;OK if not CMDR
261F 2801 00410 JR Z,NOTCMDR ;Use LOW$ otherwise
2621 04 00420 INC B
2622 00430 NOTCMDR @@HIGH$ ;P/u HIGH$/LOW$
2622 3E64 00010 LD A,100
2624 EF 00011 RST 40
2625 22D528 00440 LD (MYHIGH),HL ;Store away
2628 FDE5 00450 PUSH IY ;Trans to HL
262A D1 00460 POP DE
262B 210A00 00470 LD HL,'K'-'A' ;Offset to KFLAG$
262E 19 00480 ADD HL,DE ;HL=>KFLAG$
262F 222427 00490 LD (KFLG),HL ;Store pointer
2632 CB86 00500 RES 0,(HL) ;Kick break bit off
2634 211200 00510 LD HL,'S'-'A' ;SFLAG$ offset
2637 19 00520 ADD HL,DE
2638 22E427 00530 LD (SFLG),HL ;Store away

```

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00002
263B E1	00540	POP HL	;Restore cmd pointer
263C CD5526	00550	CALL PGRM	; and continue
	00560 ;		
	00570 ;	Exit routines	
	00580 ;		
263F 210000	00590 \$EXIT	LD HL,0	;Init to no error
2642	00600 \$QUIT	@@CKBRKC	;Clear out break bit
2642 3E6A	00012	LD A,106	
2644 EF	00013	RST 40	
2645 310000	00610	LD SP,\$-\$;P/u original stack
2646	00620 STACK	EQU \$-2	
2648 C9	00630	RET	
	00640 ;		
2649 21FFFF	00650 \$ABORT	LD HL,-1	;Set abort code
264C 18F4	00660	JR \$QUIT	; and quit
	00670 ;		
264E C5	00680 \$DSP	PUSH BC	;Display a character,
264F 4F	00690	LD C,A	; saving BC
2650	00700	@@DSP	
2650 3E02	00014	LD A,2	
2652 EF	00015	RST 40	
2653 C1	00710	POP BC	
2654 C9	00720	RET	
	00730 ;		
	00740 ;	Pick up drive numbers and partial filespec	
	00750 ;		
	00760 PGRM:		
2655 7E	00770	LD A,(HL)	;Check for NOT filespec
2656 FE2D	00780	CP '-'	; char used
2658 2006	00790	JR NZ,MVNAM1	;Go if not NOT
265A 3EFF	00800	LD A,0FFH	;TRUE value
265C 32B72A	00810	LD (NOTPRM),A	;Set if specified
265F 23	00820	INC HL	
2660 11B82A	00830 MVNAM1	LD DE,PATRN	;Point to possible partspec
2663 0608	00840	LD B,8	;Max 8 chars in name
2665 CDCF29	00850	CALL SKIPSP	;Skip spaces
2668 CDDE29	00860	CALL MOVELT	;Move letters/digits/\$
266B CDD629	00870	CALL SKIPLT	;Skip letters/digits/\$
266E 7E	00880	LD A,(HL)	;Check for extension
266F FE2F	00890	CP '/'	
2671 200C	00900	JR NZ,NOEXT	;Go if none
2673 23	00910	INC HL	
2674 11C02A	00920	LD DE,PATEXT	;Point to ext field
2677 0603	00930	LD B,3	;Max 3 chars in ext
2679 CDDE29	00940	CALL MOVELT	;Move letters/digits/\$
267C CDD629	00950	CALL SKIPLT	;Skip letters/digits/\$
267F CDBC29	00960 NOEXT	CALL GETDRV	;Get source drive #
2682 32492C	00970	LD (SDRIVE),A	;Store drive #
2685 A7	00980	AND A	;Be sure not drive 0
2686 11712B	00990	LD DE,NOT0	;Error msg
2689 EB	01000	EX DE,HL	
268A CA5529	01010	JP Z,PERR1	;Param error source is 0
268D EB	01020	EX DE,HL	;Restore cmd line ptr
268E CDCF29	01030	CALL SKIPSP	;Skip spaces
2691 CDB629	01040	CALL GETDRV2	;Get destination drive
2694 324A2C	01050	LD (DDRIVE),A	;0FFH if no dest drv
2697 CDCF29	01060	CALL SKIPSP	;Move to '('
	01070 ;		
	01080 ;	Scan parameters	
	01090 ;		
269A 11822A	01100	LD DE,PRMTBL\$;Check parameters entered

```

269D      01110      @@PARAM
269D 3E11      00016      LD      A,17
269F EF      00017      RST      40
26A0 C25129    01120      JP      NZ,PRMERR      ;Quit on parm error
26A3 210000    01130 DPARM LD      HL,$-$      ;DIR only?
26A6 7C      01140      LD      A,H
26A7 B5      01150      OR      L
26A8 2805      01160      JR      Z,SPARM      ;Go if not
26AA 3EFF      01170      LD      A,0FFH      ;Set flag at DDRIIVE
26AC 324A2C    01180      LD      (DDRIIVE),A      ;If dest is ff, read DIR
26AF 210000    01190 SPARM LD      HL,$-$      ;Check if no parms S,I,V
26B2 110000    01200 VPARM LD      DE,$-$
26B5 010000    01210 IPARM LD      BC,$-$
26B8 7D      01220      LD      A,L
26B9 B3      01230      OR      E
26BA B1      01240      OR      C
26BB 323B27    01250      LD      (SIV+1),A      ;Save S!I!V
26BE 21FFFF    01260 QPARM LD      HL,0FFFFH      ;P/u Q,N,0 parms
26C1 110000    01270 NPARM LD      DE,0
26C4 010000    01280 OPARM LD      BC,0
26C7 7B      01290      LD      A,E      ;Form N!O
26C8 B1      01300      OR      C
26C9 32F527    01310      LD      (NORO+1),A      ;Save that
      01320 ;
      01330 ;      Save old DCT
      01340 ;
26CC 3A492C    01350      LD      A,(SDRIIVE)      ;Pick up source drive #
26CF 4F      01360      LD      C,A      ;Move to C reg
26D0 3A4A2C    01370      LD      A,(DDRIIVE)      ;Be sure not single drive
26D3 B9      01380      CP      C
26D4 21462B    01390      LD      HL,NOTONE      ;=>error msg
26D7 CA5529    01400      JP      Z,PERR1      ;Go if same
26DA      01410      @GTDCT      ;Point to DCT
26DA 3E51      00018      LD      A,81
26DC EF      00019      RST      40
26DD C5      01420      PUSH     BC      ;Save drive #
26DE FDE5      01430      PUSH     IY      ;Move DCT to HL reg
26E0 E1      01440      POP      HL
26E1 11522C    01450      LD      DE,SAVDCT      ;Point to save area
26E4 010A00    01460      LD      BC,10
26E7 EDB0      01470      LDIR      ;Move it
26E9 C1      01480      POP      BC
      01490 ;
      01500 ;      Find directory track
      01510 ;
26EA 110100    01520      LD      DE,0001      ;Track 0, sector 1
26ED 21002D    01530      LD      HL,DBUFF      ;Buffer for sector
26F0      01540      @RDSEC
26F0 3E31      00020      LD      A,49
26F2 EF      00021      RST      40
26F3 2808      01550      JR      Z,OK0      ;Go if no error
26F5 FE06      01560      CP      6      ;Was it DAM error?
26F7 C23629    01570      JP      NZ,IOERR      ;Go if some other
26FA CD702A    01580      CALL     CKEARLY      ;Can we do this type?
26FD 23      01590 OK0 INC      HL      ;Point to dir cyl #
26FE 56      01600      LD      D,(HL)      ;Get it
26FF 24      01610      INC      H      ;Point to TRSDOS
2700 2B      01620      DEC      HL      ; version number
2701 2B      01630      DEC      HL
2702 2B      01640      DEC      HL
2703 7E      01650      LD      A,(HL)      ;Pick it up

```

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00004
2704 329C28	01660	LD (TRSDOS+1),A ;Save for later	
	01670 ;		
	01680 ;	Read directory records into memory	
	01690 ;		
2707 1E03	01700	LD E,3 ;Skip GAT and HIT	
2709 0610	01710	LD B,16 ;Read 16 sectors	
270B 21002D	01720	LD HL,DBUFF	
270E FD360712	01730 DREAD	LD (IY+7),18 ;Chg # sectors/trk for	
2712	01740	@@RDSEC ; TRSDOS & Read a sector	
2712 3E31	00022	LD A,49	
2714 EF	00023	RST 40	
2715 2805	01750	JR Z,OK1 ;Go if no error	
2717 FE06	01760	CP 6 ;Ignore record type	
2719 C23629	01770	JP NZ,IOERR ;Go if error	
271C 24	01780 OK1	INC H ;Bump buffer pointer	
271D 1C	01790	INC E ;Bump sector number	
271E 10EE	01800	DJNZ DREAD ;Loop till done	
	01810 ;		
	01820 ;	Loop through all entries	
	01830 ;		
2720 21002D	01840	LD HL,DBUFF ;Point to first entry	
2723	01850 ELOOP	EQU \$	
2723 3A0000	01860	LD A,(\$-\$) ;Check system break bit	
2724	01870 KFLG	EQU \$-2 ;Address of KFLAG	
2726 CB47	01880	BIT 0,A	
2728 C24926	01890	JP NZ,\$ABORT ;Abort if set	
272B 46	01900	LD B,(HL) ;P/U attributes	
272C E5	01910	PUSH HL	
272D DDE1	01920	POP IX	
272F E5	01930	PUSH HL	
2730 CB60	01940	BIT 4,B ;Alive?	
2732 CA0F29	01950	JP Z,SKIPIT ;Skip it if dead	
2735 CB78	01960	BIT 7,B ;FXDE?	
2737 C20F29	01970	JP NZ,SKIPIT ;Skip it if so	
	01980 ;		
	01990 ;	Check file's attributes	
	02000 ;		
273A 3E00	02010 SIV	LD A,\$-\$;S, I, or V given?	
273C A7	02020	AND A	
273D 2821	02030	JR Z,NOSIV ;Go if none given	
273F CB70	02040	BIT 6,B ;SYS file?	
2741 2809	02050	JR Z,NOTSYS ;Go if not	
2743 3AB026	02060	LD A,(SPARM+1) ;S parm given?	
2746 A7	02070	AND A	
2747 CA0F29	02080	JP Z,SKIPIT ;Skip file if not	
274A 1814	02090	JR NOSIV ; else possible match	
274C CB58	02100 NOTSYS	BIT 3,B ;Visible or invisible?	
274E 2009	02110	JR NZ,INV ;Go if inv	
2750 3AB326	02120	LD A,(VPARM+1) ;V parm given?	
2753 A7	02130	AND A	
2754 CA0F29	02140	JP Z,SKIPIT ;Skip file if not	
2757 1807	02150	JR NOSIV ; else possible match	
2759 3AB626	02160 INV	LD A,(IPARM+1) ;I parm given?	
275C A7	02170	AND A	
275D CA0F29	02180	JP Z,SKIPIT ;Skip file if not	
	02190 ;		
	02200 ;	Check if name matches wildcard	
	02210 ;		
2760 110500	02220 NOSIV	LD DE,5 ;Offset to name field	
2763 19	02230	ADD HL,DE	
2764 E5	02240	PUSH HL ;Compare with pattern	

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00005
2765 11B82A	02250	LD DE,PATTRN	; of user partspec
2768 060B	02260	LD B,11	
276A 1A	02270 CPLOOP	LD A,(DE)	;P/U pattern byte
276B 13	02280	INC DE	
276C FE24	02290	CP '\$'	;Matchall?
276E 2803	02300	JR Z,MATCH	
2770 BE	02310	CP (HL)	;Match?
2771 2003	02320	JR NZ,NMATCH	;Go if not
2773 23	02330 MATCH	INC HL	
2774 10F4	02340	DJNZ CPLOOP	
2776 E1	02350 NMATCH	POP HL	;Z if match, NZ if not
2777 CD0B2A	02360	CALL NOTCHK	;Reverse flag if NOT entered
277A C20F29	02370	JP NZ,SKIPIT	;Skip file if no match
	02380 ;		
277D 11092C	02390	LD DE,FCB	;Point to FCB
2780 0608	02400	LD B,8	
2782 7E	02410 MVNAME	LD A,(HL)	;Move name
2783 FE20	02420	CP ' '	;Space?
2785 2805	02430	JR Z,GOTNAM	;Go if hit one
2787 23	02440	INC HL	
2788 12	02450	LD (DE),A	;Put to FCB
2789 13	02460	INC DE	
278A 10F6	02470	DJNZ MVNAME	
278C 48	02480 GOTNAM	LD C,B	;Offset to ext field
278D 0600	02490	LD B,0	
278F 09	02500	ADD HL,BC	
2790 7E	02510	LD A,(HL)	;No extension?
2791 FE20	02520	CP ' '	
2793 2810	02530	JR Z,GOTEXT	;Go if so
2795 3E2F	02540	LD A,'/'	;Put in slash
2797 12	02550	LD (DE),A	
2798 13	02560	INC DE	
2799 0603	02570	LD B,3	
279B 7E	02580 EXLOOP	LD A,(HL)	;Move extension
279C 23	02590	INC HL	
279D FE20	02600	CP ' '	;Finished?
279F 2804	02610	JR Z,GOTEXT	
27A1 12	02620	LD (DE),A	
27A2 13	02630	INC DE	
27A3 10F6	02640	DJNZ EXLOOP	;Loop till done
	02650 ;		
27A5 3E03	02660 GOTEXT	LD A,ETX	;Put ETX at end for dsply
27A7 12	02670	LD (DE),A	
27A8 D5	02680	PUSH DE	;Save current spot in FCB
27A9 21092C	02690	LD HL,FCB	;Move name to buffer
27AC 11E92B	02700	LD DE,FNAME	; for printing
27AF 012000	02710	LD BC,32	
27B2 EDB0	02720	LDIR	
27B4 D1	02730	POP DE	;Get back where we were
	02740 ;		
	02750 ;		;Print filenames if no destination drive (DDRIVE=0FFH)
	02760 ;		
27B5 3A4A2C	02770	LD A,(DDRIVE)	;Check for just printing DIR
27B8 3C	02780	INC A	;Set Z if FF
27B9 2006	02790	JR NZ,MOVING	;Go if not FF
27BB CD1C2A	02800	CALL SHOW	;Print entry
27BE C30F29	02810	JP SKIPIT	; and go on to next
	02820 ;		
	02830 ;		Check if file exists on destination disk
	02840 ;		
27C1 3E3A	02850 MOVING	LD A,':'	;Now put the drive separator

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00006
27C3 12	02860	LD (DE),A	; in the FCB
27C4 13	02870	INC DE	
27C5 3A4A2C	02880	LD A,(DDRIVE)	;Put in drive spec
27C8 F630	02890	OR '0'	;Change number to ASCII
27CA 12	02900	LD (DE),A	
27CB 13	02910	INC DE	
27CC 3E03	02920	LD A,ETX	;Put in ETX to end
27CE 12	02930	LD (DE),A	
27CF 21092C	02940	LD HL,FCB	;Copy into 2nd FCB
27D2 11292C	02950	LD DE,FCB2	
27D5 012000	02960	LD BC,32	
27D8 EDB0	02970	LDIR	
27DA 11292C	02980	LD DE,FCB2	;Point to start of FCB
27DD 21003D	02990	LD HL,TBUFF	;Point to transfer buffer
27E0 0600	03000	LD B,0	;LRL=256
27E2 E5	03010	PUSH HL	
27E3 210000	03020	LD HL,\$-\$;HL => SFLAG
27E4	03030	EQU \$-2	
27E6 CBC6	03040	SET 0,(HL)	;Set the open inhibit bit
27E8 E1	03050	POP HL	
27E9	03060	@@OPEN	;Do the open
27E9 3E3B	00024	LD A,59	
27EB EF	00025	RST 40	
27EC 47	03070	LD B,A	;Save return code
27ED 2805	03080	JR Z,NORO	;Go if opened okay
27EF FE18	03090	CP 18H	;File not found?
27F1 C23629	03100	JP NZ,IOERR	; else an error
	03110 ;		
	03120 ;	Check New and Old parms	
	03130 ;		
27F4 3E00	03140 NORO	LD A,0	;N or 0 specified?
27F6 A7	03150	AND A	
27F7 2816	03160	JR Z,CHECKQ	;Go if neither
27F9 3AC526	03170	LD A,(OPARM+1)	;0 parm given?
27FC A7	03180	AND A	
27FD 2804	03190	JR Z,CKNEW	;Go if not
27FF AF	03200	XOR A	
2800 B0	03210	OR B	;Did file exist?
2801 280C	03220	JR Z,CHECKQ	;Go if so (ok)
2803 3AC226	03230 CKNEW	LD A,(NPARM+1)	;N parm given?
2806 A7	03240	AND A	
2807 CA0F29	03250	JP Z,SKIPIT	;Skip file if not
280A AF	03260	XOR A	
280B B0	03270	OR B	;Be sure it was new
280C CA0F29	03280	JP Z,SKIPIT	;Go if it wasn't
	03290 ;		
	03300 ;	Ask question if Q parm was given	
	03310 ;		
280F 3ABF26	03320 CHECKQ	LD A,(QPARM+1)	;Check Q parm
2812 A7	03330	AND A	
2813 2013	03340	JR NZ,QUERY	;Query if so
2815 21CA2B	03350	LD HL,CONVS	;"Converting..."
2818	03360	@@DSPLY	
	00026	IFEQ 00H,1	
	00027	LD HL,	
	00028	ENDIF	
2818 3E0A	00029	LD A,10	
281A EF	00030	RST 40	
281B 21E92B	03370	LD HL,FNAME	;Filename
281E	03380	@@DSPLY	
	00031	IFEQ 00H,1	

```

00032      LD      HL,
00033      ENDIF
281E 3E0A    00034      LD      A,10
2820 EF      00035      RST      40
2821 3E0D    03390      LD      A,CR          ;Carriage return
2823 CD4E26  03400      CALL    $DSP
2826 1841    03410      JR      TAKEIT1        ;Go & move it
          03420 ;
2828 21DC2B  03430 QUERY LD      HL,CONVQ      ;"Convert file
282B         03440 @@DSPLY  ;Display it
          00036      IFEQ    00H,1
          00037      LD      HL,
          00038      ENDIF
282B 3E0A    00039      LD      A,10
282D EF      00040      RST      40
282E 21A92B  03450      LD      HL,QMARK      ;"?"
2831         03460 @@DSPLY
          00041      IFEQ    00H,1
          00042      LD      HL,
          00043      ENDIF
2831 3E0A    00044      LD      A,10
2833 EF      00045      RST      40
2834 214D2C  03470      LD      HL,ABUFF      ;Get answer
2837 010003  03480      LD      BC,3<8        ;3 char max
283A         03490 @@KEYIN
283A 3E09    00046      LD      A,9
283C EF      00047      RST      40
283D DA4926  03500      JP      C,$ABORT      ;Abort if BREAK hit
2840 7E      03510      LD      A,(HL)      ;Check for 'Y'
2841 CBAF    03520      RES     5,A          ;Force upper case
2843 FE59    03530      CP      'Y'
2845 C20F29  03540      JP      NZ,SKIPIT      ;Skip it if not 'Y'
          03550 ;
          03560 ;      If file exists, query user
          03570 ;
2848 3A292C  03580      LD      A,(FCB2)      ;Was file opened ok?
284B CB7F    03590      BIT     7,A          ;Z = not found
284D 281A    03600      JR      Z,TAKEIT1      ;Go if it does not exist
284F 21AC2B  03610      LD      HL,EXISTQ      ;"File exists, replace?
2852         03620 @@DSPLY  ;Print question
          00048      IFEQ    00H,1
          00049      LD      HL,
          00050      ENDIF
2852 3E0A    00051      LD      A,10
2854 EF      00052      RST      40
2855 214D2C  03630      LD      HL,ABUFF
2858 010003  03640      LD      BC,3<8
285B         03650 @@KEYIN          ;Get answer
285B 3E09    00053      LD      A,9
285D EF      00054      RST      40
285E DA4926  03660      JP      C,$ABORT      ;Abort if break
2861 7E      03670      LD      A,(HL)      ;Check answer
2862 CBAF    03680      RES     5,A          ;Force uppercase
2864 FE59    03690      CP      'Y'
2866 C20F29  03700      JP      NZ,SKIPIT      ;Skip if 'no'
          03710 ;
          03720 ;      Init file if it didn't exist
          03730 ;
2869 11292C  03740 TAKEIT1 LD      DE,FCB2
286C 1A      03750      LD      A,(DE)      ;Was file opened?
286D CB7F    03760      BIT     7,A          ;Z = not opened

```

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00008
286F 2803	03770	JR Z,\$+5	;Remove existing file
2871	03780	@@REMOV	; for new LRL
2871 3E39	00055	LD A,57	
2873 EF	00056	RST 40	
2874 11092C	03790	LD DE,FCB	;Use other FCB now
2877 21003D	03800	LD HL,TBUFF	;Create file
287A DD4604	03810	LD B,(IX+4)	;P/U Mod III LRL
287D	03820	@@INIT	;Create the file
287D 3E3A	00057	LD A,58	
287F EF	00058	RST 40	
2880 C23629	03830	JP NZ,IOERR	;Go if error
2883 D5	03840	PUSH DE	;Change LRL to 0 for copy
2884 DDE3	03850	EX (SP),IX	;IX to FCB start
2886 DDCB01BE	03860	RES 7,(IX+1)	;Show full sector ops
288A DD360900	03870	LD (IX+9),0	;Show LRL=0
288E DDE3	03880	EX (SP),IX	;Switch back
2890 D1	03890	POP DE	
	03900 ;		
	03910 ;		
	03920 ;		
	03930	Initialize to read from source file	
2891 E1	03930 TAKEIT2	POP HL	;Point to dir entry
2892 E5	03940	PUSH HL	
2893 111400	03950	LD DE,20	;Point to ERN
2896 19	03960	ADD HL,DE	
2897 5E	03970	LD E,(HL)	;P/U ERN
2898 23	03980	INC HL	
2899 56	03990	LD D,(HL)	
289A 23	04000	INC HL	;Leave ptg to extents
289B 3E00	04010 TRSDOS	LD A,0	;1.3 or later?
289D FE13	04020	CP 13H	
289F 3807	04030	JR C,EARLY	;Go if earlier than 1.3
28A1 DD7E03	04040	LD A,(IX+3)	;Pick up EOF offset
28A4 A7	04050	AND A	;Zero?
28A5 2801	04060	JR Z,EARLY	;No adjustment if so
28A7 13	04070	INC DE	;If nonzero, adjust ERN
28A8 0600	04080 EARLY	LD B,0	;# sectors left in extent
28AA D5	04090	PUSH DE	;Save ERN
28AB D9	04100	EXX	;Switch to alternate regs
	04110 ;		
	04120 ;		
	04130 ;		
	04140	Preallocate file	
28AC C1	04140	POP BC	
28AD 78	04150	LD A,B	;Empty file?
28AE B1	04160	OR C	
28AF 281E	04170	JR Z,READ	;Go if so
28B1 0B	04180	DEC BC	
28B2 11092C	04190	LD DE,FCB	;Point to FCB
28B5	04200	@@POSN	;Position to last sector
28B5 3E42	00059	LD A,66	
28B7 EF	00060	RST 40	
28B8 2809	04210	JR Z,OK3	
28BA FE1C	04220	CP 1CH	;Ignore EOF errors
28BC 2805	04230	JR Z,OK3	
28BE FE1D	04240	CP 1DH	; or past end errors
28C0 C23629	04250	JP NZ,IOERR	;Quit on any others
28C3	04260 OK3	@@WRITE	;Write it
28C3 3E4B	00061	LD A,75	
28C5 EF	00062	RST 40	
28C6 C23629	04270	JP NZ,IOERR	;Quit on write error
28C9	04280	@@REW	;Position to start
28C9 3E44	00063	LD A,68	

```

28CB EF      00064      RST      40
28CC C23629  04290      JP       NZ,IOERR
                04300 ;
                04310 ;      Read sectors
                04320 ;
28CF 0600    04330 READ   LD       B,0      ;Count sectors read
28D1 21003D  04340      LD       HL,TBUFF  ;Point to transfer buffer
28D4 110000  04350      LD       DE,$-$
28D5         04360 MYHIGH EQU      $-2      ;Stuff HIGH$ value
28D7 15      04370      DEC      D          ;256 bytes back
28D8 CD5B29  04380 GETONE CALL    GETSEC   ;Get next sector
28DB 200B    04390      JR       NZ,WRITE  ;Go if EOF
28DD 04      04400      INC      B          ;Count sector
28DE 24      04410      INC      H          ;Point to next spot
28DF CD052A  04420      CALL    CPHLDE    ;Compare HL and DE
28E2 3E00    04430      LD       A,0      ;No error code
28E4 3002    04440      JR       NC,WRITE  ;Go if mem full
28E6 18F0    04450      JR       GETONE   ; else loop for more
                04460 ;
                04470 ;      Write sectors to destination file
                04480 ;
28E8 F5      04490 WRITE  PUSH     AF      ;Save completion type
28E9 11092C  04500      LD       DE,FCB    ;Point to file fcb
28EC 21003D  04510      LD       HL,TBUFF  ;Point to transfer buffer
28EF 220C2C  04520 WRLOOP LD      (FCB+3),HL ;Point FCB to buffer
28F2 78      04530      LD       A,B      ;Zero to write?
28F3 A7      04540      AND      A
28F4 2809    04550      JR       Z,WRDUN   ;Go if so
28F6         04560      @@WRITE
28F6 3E4B    00065      LD       A,75
28F8 EF      00066      RST      40
28F9 C23629  04570      JP       NZ,IOERR  ;Quit on write error
28FC 24      04580      INC      H
28FD 10F0    04590      DJNZ     WRLOOP   ;Loop till done
                04600 ;
                04610 ;      Were we at EOF?
                04620 ;
28FF F1      04630 WRDUN  POP      AF      ;Restore completion type
2900 A7      04640      AND      A          ;At end of file?
2901 28CC    04650      JR       Z,READ    ;Go if not
                04660 ;
                04670 ;      Copy over EOF offset
                04680 ;
2903 DD7E03  04690      LD       A,(IX+3)  ;P/U offset from dir
2906 32112C  04700      LD       (FCB+8),A ;Put into FCB
2909         04710      @@CLOSE          ; and close the file
2909 3E3C    00067      LD       A,60
290B EF      00068      RST      40
290C C23629  04720      JP       NZ,IOERR  ;Quit on close error
                04730 ;
                04740 ;      Increment to next entry and loop if not done
                04750 ;
290F E1      04760 SKIPIT POP      HL
2910 113000  04770      LD       DE,48    ;48 bytes per entry
2913 19      04780      ADD      HL,DE
2914 7D      04790      LD       A,L      ;End of sector?
2915 FEF0    04800      CP       0FH
2917 2003    04810      JR       NZ,NOTEOS ;Go if not
2919 24      04820      INC      H
291A 2E00    04830      LD       L,0
291C 11003D  04840 NOTEOS LD      DE,TBUFF ;Done?

```

```

291F CD052A 04850 CALL CPHLDE ;CP HL,DE
2922 DA2327 04860 JP C,ELOOP ;Loop back if not done
          04870 ;
          04880 ; Finished
          04890 ;
2925 3E0D 04900 LD A,CR ;Blank line
2927 CD4E26 04910 CALL $DSP
292A CD4529 04920 CALL BYEBYE ;Restore DCT
292D C33F26 04930 JP $EXIT
          04940 ;
2930 CD4529 04950 QUIT CALL BYEBYE ;Restore DCT
2933 C34926 04960 JP $ABORT
          04970 ;
          04980 ; Error routines
          04990 ;
2936 CD4529 05000 IOERR CALL BYEBYE ;Restore DCT
2939 6F 05010 IOERR1 LD L,A ;Entry from PRMERR
293A 2600 05020 LD H,0
293C F6C0 05030 OR 0C0H ;Abbrev, return
293E 4F 05040 LD C,A ;Error code to C
293F 05050 @@ERROR ; for error display
293F 3E1A 00069 LD A,26
2941 EF 00070 RST 40
2942 C34226 05060 JP $QUIT
          05070 ;
2945 FDE5 05080 BYEBYE PUSH IY ;Move back DCT
2947 D1 05090 POP DE
2948 21522C 05100 LD HL,SAVDCT ;Point to save area
294B 010A00 05110 LD BC,10
294E EDB0 05120 LDIR
2950 C9 05130 RET
          05140 ;
2951 3E2C 05150 PRMERR LD A,44 ;Init "parameter error
2953 18E4 05160 JR IOERR1
2955 05170 PERR1 @@LOGOT ;Display and log
          00071 IFEQ 00H,1
          00072 LD HL,
          00073 ENDIF
2955 3E0C 00074 LD A,12
2957 EF 00075 RST 40
2958 C34926 05180 JP $ABORT
          05190 ;
          05200 ; Sector read routine
          05210 ;
295B D9 05220 GETSEC EXX ;P/U alt registers
295C 7A 05230 LD A,D ;Any records left?
295D B3 05240 OR E
295E 2005 05250 JR NZ,NOTEND ;Go if so
2960 D9 05260 BDEXT EXX
2961 3E1C 05270 LD A,1CH ;EOF code
2963 A7 05280 AND A ;Set NZ condition
2964 C9 05290 RET
          05300 ;
2965 AF 05310 NOTEND XOR A ;Check if used up ext
2966 B0 05320 OR B
2967 2021 05330 JR NZ,MORE ;Go if not used up
2969 7E 05340 LD A,(HL) ;Check next trk#
296A FEFF 05350 CP 0FFH ;Non-allocated?
296C 28F2 05360 JR Z,BDEXT ;Then consider EOF
296E D5 05370 PUSH DE ;Save DE'
296F 56 05380 LD D,(HL) ;P/U track number

```

```

2970 23      05390      INC      HL
2971 46      05400      LD        B,(HL)      ;P/U other stuff
2972 23      05410      INC      HL
2973 78      05420      LD        A,B          ;Get starting gran
2974 07      05430      RLCA
2975 07      05440      RLCA          ;Move to bits 0-2
2976 07      05450      RLCA
2977 E607    05460      AND        7          ;Mask off other garbage
2979 5F      05470      LD        E,A        ;Multiply by 3
297A 07      05480      RLCA
297B 83      05490      ADD        A,E
297C 3C      05500      INC        A          ;Offset from 0
297D 5F      05510      LD        E,A        ; and move to E reg
297E ED534B2C 05520      LD        (TRKSEC),DE ;Save for later
2982 D1      05530      POP        DE        ;Restore DE'
2983 78      05540      LD        A,B          ;Get number of grans
2984 E61F    05550      AND        1FH
2986 47      05560      LD        B,A        ;Multiply by 3
2987 07      05570      RLCA
2988 80      05580      ADD        A,B
2989 47      05590      LD        B,A        ;And put in B reg
          05600      ;
          05610      ;      Read sector
          05620      ;
          05630      ;
298A 05      05630      MORE      DEC        B          ;Count down # sec in ext
298B 1B      05640      DEC        DE        ;Count down # records
298C D9      05650      EXX          ;Restore primary set
298D D5      05660      PUSH       DE        ;Save DE
298E C5      05670      PUSH       BC        ;Save BC
298F ED5B4B2C 05680      LD        DE,(TRKSEC) ;P/U track and sector #
2993 3A492C  05690      LD        A,(SDRIVE) ;P/U source drive
2996 4F      05700      LD        C,A
2997 FD360712 05710      LD        (IY+7),18 ;Reset sec/trk each time
299B        05720      @@RDSEC ;Read sector to (HL)
299B 3E31    05730      LD        A,49
299D EF      05740      RST        40
299E 2805    05750      JR        Z,OK2      ;Go if no errors
29A0 FE06    05760      CP        6          ; or address mark differs
29A2 C23629 05770      JP        NZ,IOERR    ;Quit on any other
29A5 1C      05780      INC        E        ;Step to next sector
29A6 7B      05790      LD        A,E
29A7 FE13    05800      CP        19D      ;End of track?
29A9 2003    05810      JR        NZ,NOTEOT ;Go if not
29AB 1E01    05820      LD        E,1      ;Reset to sector 1
29AD 14      05830      INC        D        ;Next track
29AE ED534B2C 05840      NOTEOT  LD        (TRKSEC),DE
29B2 C1      05850      POP        BC
29B3 D1      05860      POP        DE
29B4 AF      05870      XOR        A
29B5 C9      05880      RET
          05890      ;
          05900      ;      Parsing subroutines
          05910      ;
29B6 7E      05920      GETDRV2 LD        A,(HL)
29B7 FE3A    05930      CP        ':'
29B9 3EFF    05940      LD        A,0FFH    ;'Not entered' value
29BB C0      05950      RET        NZ      ;If no second drive, give DIR
          05960      ;
29BC 7E      05970      GETDRV  LD        A,(HL) ;Parse drivespec
29BD FE3A    05980      CP        ':'
29BF 2090    05990      JR        NZ,PRMERR ;Go if missing

```

```

29C1 23      05980      INC      HL
29C2 7E      05990      LD        A,(HL)      ;P/U drivespec
29C3 FE30    06000      CP        '0'      ;Be sure digit
29C5 388A    06010      JR        C,PRMERR
29C7 FE38    06020      CP        '7'+1
29C9 3086    06030      JR        NC,PRMERR
29CB 23      06040      INC      HL      ;Bump cmdline ptr
29CC E607    06050      AND      7      ;Make drive # binary
29CE C9      06060      RET
                06070 ;
29CF 7E      06080 SKIPSP LD        A,(HL)      ;Skip spaces
29D0 FE20    06090      CP        ','
29D2 C0      06100      RET      NZ
29D3 23      06110      INC      HL
29D4 18F9    06120      JR        SKIPSP
                06130 ;
29D6 7E      06140 SKIPLT LD        A,(HL)      ;Skip letters/digits/$
29D7 CDE829 06150      CALL    CHKLET    ;Check letter/digit/$
29DA C0      06160      RET      NZ
29DB 23      06170      INC      HL
29DC 18F8    06180      JR        SKIPLT
                06190 ;
29DE 7E      06200 MOVELT LD        A,(HL)      ;Move letters/digits/$
29DF CDE829 06210      CALL    CHKLET
29E2 C0      06220      RET      NZ
29E3 23      06230      INC      HL      ;Inc from buffer
29E4 12      06240      LD        (DE),A    ;Store
29E5 13      06250      INC      DE      ;Inc to buffer
29E6 18F6    06260      JR        MOVELT
                06270 ;
29E8 CB7F    06280 CHKLET BIT      7,A      ;Graphic?
29EA C0      06290      RET      NZ
29EB FE61    06300      CP        'a'      ;Lowercase?
29ED 3802    06310      JR        C,NOTLC    ;Go if not
29EF CBAF    06320      RES      5,A      ; else make upper case
29F1 FE24    06330 NOTLC  CP        '$'      ;Dollar sign?
29F3 C8      06340      RET      Z
29F4 FE30    06350      CP        '0'      ;Digit?
29F6 D8      06360      RET      C      ;Return (NZ) if less
29F7 FE3A    06370      CP        '9'+1
29F9 3002    06380      JR        NC,NOTDIG    ;Go if not digit
29FB BF      06390      CP        A      ;Mark as letter/digit/$
29FC C9      06400      RET
29FD FE41    06410 NOTDIG CP      'A'      ;Letter?
29FF D8      06420      RET      C      ;Return (NZ) if less
2A00 FE5A    06430      CP        'Z'
2A02 D0      06440      RET      NC      ;Z if =Z, NZ if >Z
2A03 BF      06450      CP        A      ;Z if <Z
2A04 C9      06460      RET
                06470 ;
2A05 E5      06480 CPHLDE PUSH    HL      ;Compare HL and DE
2A06 A7      06490      AND      A
2A07 ED52    06500      SBC      HL,DE
2A09 E1      06510      POP      HL
2A0A C9      06520      RET
                06530 ;
                06540 ;If NOT (-) spec given, reverse Z flag setting
                06550 ;
2A0B F5      06560 NOTCHK PUSH    AF      ;Save current setting
2A0C 3AB72A 06570      LD        A,(NOTPRM) ;Was NOT entered?
2A0F B7      06580      OR        A

```



```

2A10 2808    06590    JR      Z,NOTNOT    ;No, restore previous
2A12 F1      06600    POP      AF          ;Get previous
2A13 2802    06610    JR      Z,SETIT    ;Was Z, make NZ
2A15 AF      06620    XOR      A          ; else was NZ, make Z
2A16 C9      06630    RET
2A17 F6FF    06640 SETIT  OR      0FFH      ;make NZ
2A19 C9      06650    RET
2A1A F1      06660 NOTNOT POP      AF          ;Get previous flags
2A1B C9      06670    RET
                06680 ;
                06690 ;Display mod 3 TRSDOS disk directory
                06700 ;
2A1C E5      06710 SHOW  PUSH    HL
2A1D D5      06720      PUSH    DE
2A1E C5      06730      PUSH    BC          ;Save registers
2A1F 0E00    06740      LD      C,0        ;Init char count
2A21 21E92B  06750      LD      HL,FNAME    ;=>name
2A24 7E      06760 NMDSP LD      A,(HL)    ;Get a character
2A25 FE03    06770      CP      ETX        ;Are we done?
2A27 2807    06780      JR      Z,NMEND    ;Finish if so
2A29 CD4E26  06790      CALL   $DSP       ;Print this char
2A2C 0C      06800      INC      C          ;Count it
2A2D 23      06810      INC      HL        ;=>next char
2A2E 18F4    06820      JR      NMDSP      ;Until ETX
                06830 ;
2A30 210000  06840 NMEND  LD      HL,$-$    ;P/u line/char count
2A31         06850 CCOUNT EQU    $-2
2A33 79      06860      LD      A,C        ;Count for this entry
2A34 85      06870      ADD      A,L        ;Add to previous
2A35 6F      06880      LD      L,A        ;Save posn
2A36 3E10    06890      LD      A,16       ;Spaces for entry
2A38 91      06900      SUB      C          ;Less used
2A39 47      06910      LD      B,A        ;Remaining to B
2A3A 3E20    06920 SPLP  LD      A,' '     ;Pad remaining w/spaces
2A3C CD4E26  06930      CALL   $DSP
2A3F 2C      06940      INC      L          ;Count it
2A40 7D      06950      LD      A,L        ;Check char posn
2A41 FE4E    06960      CP      78        ;End of line?
2A43 2809    06970      JR      Z,ELINE    ;Then print CR
2A45 10F3    06980      DJNZ    SPLP      ; else keep going
                06990 ;
2A47 22312A  07000 ESHOW LD      (CCOUNT),HL ;Save line/char posn
2A4A C1      07010      POP      BC        ;Restore regs
2A4B D1      07020      POP      DE
2A4C E1      07030      POP      HL
2A4D C9      07040      RET              ;Done w/entry
                07050 ;
2A4E 3E0D    07060 ELINE LD      A,CR      ;Hit end of line
2A50 CD4E26  07070      CALL   $DSP
2A53 24      07080      INC      H          ;Bump line posn
2A54 2E00    07090      LD      L,0        ;Start on next
2A56 3E17    07100      LD      A,23       ;Max lines
2A58 BC      07110      CP      H          ;There yet?
2A59 20EC    07120      JR      NZ,ESHOW   ;Nope
2A5B         07130      @KEY             ;Wait for a key
2A5B 3E01    00078      LD      A,1
2A5D EF      00079      RST      40
2A5E CD662A  07140      CALL   $CLS      ;Clear the display
2A61 210000  07150      LD      HL,0        ;Restart count
2A64 18E1    07160      JR      ESHOW
                07170 ;

```

The Source	UTILITY Files	CONV - LS-DOS 6.2	Page 00014
2A66 3E1C	07180 \$CLS LD	A,HOME ;Cursor home	
2A68 CD4E26	07190 CALL	\$DSP	
2A6B 3E1F	07200 LD	A,CLR ;Clear to end-of-frame	
2A6D C34E26	07210 JP	\$DSP	
	07220 ;		
2A70 00	07230 CKEARLY DB	0	
2A71 3A222D	07240 LD	A,(DBUFF+22H) ;Get type byte	
2A74 FEFF	07250 CP	0FFH ;Do we know this one?	
2A76 C8	07260 RET	Z ;OK to continue	
2A77 3A4A2C	07270 LD	A,(DDRIVE) ;Doesn't matter if	
2A7A 3C	07280 INC	A ; only doing DIR	
2A7B C8	07290 RET	Z	
2A7C 218A2B	07300 LD	HL,EARLYD ;Err msg	
2A7F C35529	07310 JP	PERR1 ;Quit	
	07320 ;		
2A82 80	07330 PRMTBL\$ DB	80H	
2A83 55	07340 DB	ABB!FLAG!5	
2A84 51	07350 DB	'QUERY',0	
	55 45 52 59 00		
2A8A BF26	07360 DW	QPARM+1	
2A8C 53	07370 DB	ABB!FLAG!3	
2A8D 53	07380 DB	'SYS',0	
	59 53 00		
2A91 B026	07390 DW	SPARM+1	
2A93 53	07400 DB	ABB!FLAG!3	
2A94 49	07410 DB	'INV',0	
	4E 56 00		
2A98 B626	07420 DW	IPARM+1	
2A9A 53	07430 DB	ABB!FLAG!3	
2A9B 56	07440 DB	'VIS',0	
	49 53 00		
2A9F B326	07450 DW	VPARM+1	
2AA1 53	07460 DB	ABB!FLAG!3	
2AA2 4F	07470 DB	'OLD',0	
	4C 44 00		
2AA6 C526	07480 DW	OPARM+1	
2AA8 53	07490 DB	ABB!FLAG!3	
2AA9 4E	07500 DB	'NEW',0	
	45 57 00		
2AAD C226	07510 DW	NPARM+1	
2AAF 53	07520 DB	ABB!FLAG!3	
2AB0 44	07530 DB	'DIR',0	
	49 52 00		
2AB4 A426	07540 DW	DPARM+1	
2AB6 00	07550 NOP		
	07560 ;		
	07570 ;	Messages and buffers	
	07580 ;		
2AB7 00	07590 NOTPRM DB	0	
2AB8 24	07600 PATTRN DB	'\$\$\$\$\$\$\$'	
	24 24 24 24 24 24		
2AC0 24	07610 PATEXT DB	'\$\$\$'	
	24 24		
2AC3 43	07620 HELLO\$ DB	'CONV'	
	4F 4E 56		
2AC7	07630 *GET CLIENT:3		
	03950 ;CLIENTS/ASM - File to establish sign-on headers		
	03960 ;		
2AC7 20	03970 DB	' - 6.2.0 - Copyright 1982/83/84 by Logical'	
	2D 20 36 2E 32 2E 30 20		
	2D 20 43 6F 70 79 72 69		
	67 68 74 20 31 39 38 32		

```

2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61
6C
2AF1 20      03980      DB      ' Systems, Inc.      ',10
53 79 73 74 65 6D 73 2C
20 49 6E 63 2E 20 20 20
20 20 20 0A
      03990 ;
2B06 41      04000      DB      'All Rights Reserved. Licensed 1982/83/84'
6C 6C 20 52 69 67 68 74
73 20 52 65 73 65 72 76
65 64 2E 20 4C 69 63 65
6E 73 65 64 20 31 39 38
32 2F 38 33 2F 38 34
2B2E 20      04010      DB      ' to xxxxxxxxxxxxxxxxxxxx',10,13
74 6F 20 78 78 78 78 78
78 78 78 78 78 78 78 78
78 78 78 78 78 0A 0D
2B46 53      07640 NOTONE DB      'Source and Destination drives are the same',CR
6F 75 72 63 65 20 61 6E
64 20 44 65 73 74 69 6E
61 74 69 6F 6E 20 64 72
69 76 65 73 20 61 72 65
20 74 68 65 20 73 61 6D
65 0D
2B71 53      07650 NOT0   DB      'Source cannot be drive 0',CR
6F 75 72 63 65 20 63 61
6E 6E 6F 74 20 62 65 20
64 72 69 76 65 20 30 0D
2B8A 43      07660 EARLYD DB      'Cannot CONV Protected Diskette',CR
61 6E 6E 6F 74 20 43 4F
4E 56 20 50 72 6F 74 65
63 74 65 64 20 44 69 73
6B 65 74 74 65 0D
2BA9 3F      07670 QMARK  DB      '? ',ETX
20 03
2BAC 20      07680 EXISTQ DB      ' File exists -- replace it? ',ETX
20 46 69 6C 65 20 65 78
69 73 74 73 20 2D 2D 20
72 65 70 6C 61 63 65 20
69 74 3F 20 03
2BCA 43      07690 CONVS  DB      'Converting file: ',ETX
6F 6E 76 65 72 74 69 6E
67 20 66 69 6C 65 3A 20
03
2BDC 43      07700 CONVQ  DB      'Convert file '
6F 6E 76 65 72 74 20 66
69 6C 65 20
0020      07710 FNAME   DS      32      ;Must follow CONVQ
0020      07720 FCB     DS      32      ;For INIT/WRITE
0020      07730 FCB2    DS      32      ;For OPEN (test for already existing)
0001      07740 SDRIVE  DS      1
0001      07750 DDRIVE  DS      1
0002      07760 TRKSEC  DS      2
0005      07770 ABUFF   DS      5
000A      07780 SAVDCT  DS      10
2D00      07790         ORG      $<-8+1<8
1000      07800 DBUFF   DS      1000H    ;16 sectors of directory
3D00      07810 TBUFF   EQU      $      ;To end of memory
2600      07820         END      BEGIN

```

\$ABORT	2649	\$CLS	2A66	\$DSP	264E
\$EXIT	263F	\$QUIT	2642	@@1	0000
@@2	0000	@@3	0000	@@4	0000
@MOD2	0000	@MOD4	FFFF	ABB	0010
ABUFF	2C4D	BDEXT	2960	BEGIN	2600
BEGINA	2609	BYEBYE	2945	CCOUNT	2A31
CHECKQ	280F	CHKLET	29E8	CKEARLY	2A70
CKNEW	2803	CLR	001F	CONVQ	2BDC
CONVS	2BCA	CPHLDE	2A05	CPLLOOP	276A
CR	000D	DBUFF	2D00	DDRIVE	2C4A
DPARM	26A3	DREAD	270E	EARLY	28A8
EARLYD	2B8A	ELINE	2A4E	ELOOP	2723
ESHOW	2A47	ETX	0003	EXISTQ	2BAC
EXLOOP	279B	FCB	2C09	FCB2	2C29
FLAG	0040	FNAME	2BE9	GETDRV	29BC
GETDRV2	29B6	GETONE	28D8	GETSEC	295B
GOTEXT	27A5	GOTNAM	278C	HELLO\$	2AC3
HOME	001C	INV	2759	IOERR	2936
IOERR1	2939	IPARM	26B5	KFLG	2724
LF	000A	MATCH	2773	MORE	298A
MOVELT	29DE	MOVING	27C1	MVNAM1	2660
MVNAME	2782	MYHIGH	28D5	NMATCH	2776
NMDSP	2A24	NMEND	2A30	NOEXT	267F
NORO	27F4	NOSIV	2760	NOT0	2B71
NOTCHK	2A0B	NOTCMDR	2622	NOTDIG	29FD
NOTEND	2965	NOTEOS	291C	NOTEOT	29AE
NOTLC	29F1	NOTNOT	2A1A	NOTONE	2B46
NOTPRM	2AB7	NOTSYS	274C	NPARAM	26C1
OK0	26FD	OK1	271C	OK2	29A5
OK3	28C3	OPARM	26C4	PATEXT	2AC0
PATRN	2AB8	PERR1	2955	PGRM	2655
PRMERR	2951	PRMTBL\$	2A82	QMARK	2BA9
QPARM	26BE	QUERY	2828	QUIT	2930
READ	28CF	SAVDCT	2C52	SDRIVE	2C49
SETIT	2A17	SFLG	27E4	SHOW	2A1C
SIV	273A	SKIPIT	290F	SKIPLT	29D6
SKIPSP	29CF	SPARM	26AF	SPLP	2A3A
STACK	2646	TAKEIT1	2869	TAKEIT2	2891
TBUFF	3D00	TRKSEC	2C4B	TRSDOS	289B
VPARM	26B2	WRDUN	28FF	WRITE	28E8
WRLOOP	28EF	@@ABORT	B00E	@@ADTSK	B0A1
@@BANK	B5B9	@@BKSP	B299	@@BREAK	B5CF
@@CHNIO	AFF9	@@CKBRKC	B61D	@@CKDRV	B0F5
@@CKEOF	B2AE	@@CKTSK	B08C	@@CLOSE	B284
@@CLS	B607	@@CMNDI	B038	@@CMNDR	B04D
@@CTL	AE5D	@@DATE	AFCF	@@DCSTAT	B134
@@DEBUG	B077	@@DECHEX	B539	@@DIRRD	B4A6
@@DIRWR	B4BB	@@DIV16	B524	@@DIV8	B50F
@@DODIR	B10A	@@DSP	AE21	@@DSPLY	AEC1
@@ERROR	B062	@@EXIT	B023	@@FEXT	B413
@@FLAGS	B5A3	@@FNAME	B428	@@FSPEC	B3FE
@@GATRD	B491	@@GATWR	B4D0	@@GET	AE35
@@GTDCB	B452	@@GTDCI	B43D	@@GTMOD	B467
@@HDFMT	B1DC	@@HEX16	B578	@@HEX8	B563
@@HEXDEC	B54E	@@HIGH\$	B58D	@@INIT	B25A
@@KBD	AE99	@@KEY	AE0D	@@KEYIN	AEAD
@@KLTsk	B0E0	@@LOAD	B3D4	@@LOC	B2C3
@@LOF	B2D8	@@LOGGER	AEF8	@@LOGOT	AF0D
@@MSG	AF44	@@MUL16	B4FA	@@MUL8	B4E5
@@OPEN	B26F	@@PARAM	AFBA	@@PAUSE	AF45

@@PEOF	B2ED @@POSN	B302 @@PRINT	AF59
@@PRT	AE71 @@PUT	AE49 @@RAMDIR	B11F
@@RDSEC	B1B2 @@RDSSC	B47C @@READ	B317
@@REMOV	B245 @@RENAM	B230 @@REW	B32C
@@RMTSK	B0B6 @@RPTSK	B0CB @@RREAD	B341
@@RSLCT	B19D @@RSTOR	B15E @@RUN	B3E9
@@RWIT	B356 @@SEEK	B188 @@SEEKSC	B36B
@@SKIP	B380 @@SLCT	B149 @@STEPI	B173
@@TIME	AFE4 @@VDCTL	AF90 @@VER	B395
@@VRSEC	B1C7 @@WEOF	B3AA @@WHERE	AE85
@@WRITE	B3BF @@WRSEC	B1F1 @@WRSSC	B206
@@WRTRK	B21B		
00000 Total errors			

NOTES:

NOTES:

FLOPPY/DCT - 5 1/4" drive setup

The Floppy DCT program allows up to four physical 5 1/4" floppy drives to be assigned to the seven different logical drive positions. It is activated with the SYSTEM (DRIVER) Library command.


```

00100 ;LDOSDCT/ASM - Floppy Disk DCT
0000 00110 TITLE <FLOPPY/DCT - LS-DOS 6.2>
00120 ;
00130 ; Program installs a standard DCT into a logical
00140 ; drive slot as specified by:
00150 ; SYSTEM (DRIVE=d,DRIVER="LDOS")
00160 ; The default DCT is taken from slot 0 of the
00170 ; System Information Sector (70H-79H).
00180 ;
000D 00190 CR EQU 13
000A 00200 LF EQU 10
00210 ;
0000 00220 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
0000 00230 *GET COPYCOM:3 ;Copyright message
00320 ; COPYCOM - File for Copyright COMMENT block
00330 ;
0000 00340 COM '<*(C) 1982,83,84 by LSI*>'
00240 ;
2C00 00250 ORG 2C00H
00260 ;
00270 BEGIN
2C00 00280 @@CKBRKC
2C00 3E6A 00001 LD A,106
2C02 EF 00002 RST 40
2C03 2804 00290 JR Z,BEGINA ;Continue if no break
2C05 21FFFF 00300 LD HL,-1 ; else abort
2C08 C9 00310 RET
00320 ;
2C09 D5 00330 BEGINA PUSH DE ;Save the DCT location
2C0A 00340 @@DSPLY HELLO$ ;Display the signon
00003 IFEQ 01H,1
2C0A 21E22C 00004 LD HL,HELLO$
00005 ENDIF
2C0D 3E0A 00006 LD A,10
2C0F EF 00007 RST 40
2C10 D1 00350 POP DE
2C11 7A 00360 LDOS LD A,D ;Make sure that a
2C12 B3 00370 OR E ; drive # was entered
2C13 CAD52C 00380 JP Z,NODRV ;Go if no drive
00390 ;
00400 ; Check if entry from SET command
00410 ;
2C16 00420 @@FLAGS
2C16 3E65 00008 LD A,101
2C18 EF 00009 RST 40
2C19 FDCB025E 00000 BIT 3,(IY+'C'-'A') ;System request?
2C1D CAC92C 00440 JP Z,VIASET ;Exit if not
2C20 1A 00450 LD A,(DE)
2C21 FEC9 00460 CP 0C9H ;Is drive disabled?
2C22 C2CD 00470 JP NZ,ACTIVE ;Must be disabled
2C26 D5 00480 PUSH DE ;Save DCT address
2C27 CDB22C 00490 CALL GETCFG ;Load sysinfo sector
2C2A C2BD2C 00500 JP NZ,IOERR ;Quit on read error
2C2D FDCB0B66 00510 BIT 4,(IY+'L'-'A') ;Suppress 8" queries?
2C31 201C 00520 JR NZ,LDOS3 ;NZ=suppress
00530 ;
00540 ; Query as to 5" or 8" floppy
00550 ;

```

```

2C33 21DB2D 00560 DRV TYP LD HL,DRV TYP$ ;"Enter drive code...
2C36 00570 @@DSPLY
00010 IFEQ 00H,1
00011 LD HL,
00012 ENDIF
2C36 3E0A 00013 LD A,10
2C38 EF 00014 RST 40
2C39 21312E 00580 LD HL,BUF ;Pt to buffer
2C3C 010001 00590 LD BC,1<8 ;Allow 1 char only
2C3F 00600 @@KEYIN ;Get response
2C3F 3E09 00015 LD A,9
2C41 EF 00016 RST 40
2C42 DAD12C 00610 JP C,BREAK ;Quit on Break
2C45 7E 00620 LD A,(HL) ;P/u char response
2C46 D630 00630 SUB '0' ;Adjust to binary
2C48 FE02 00640 CP 2 ;Make sure requested
2C4A 30E7 00650 JR NC,DRV TYP ; type is supported
2C4C 32722C 00660 LD (LX805+1),A
00670 ;
00680 ; Prompt user for physical drive address
00690 ;
2C4F 00700 LDOS3 @@DSPLY DRVADR$ ;"Enter physical...
00017 IFEQ 01H,1
2C4F 21FF2D 00018 LD HL,DRVADR$
00019 ENDIF
2C52 3E0A 00020 LD A,10
2C54 EF 00021 RST 40
2C55 21312E 00710 LD HL,BUF ;Input buffer
2C58 010001 00720 LD BC,1<8 ;Allow 1 char only
2C5B 00730 @@KEYIN ;Get response
2C5B 3E09 00022 LD A,9
2C5D EF 00023 RST 40
2C5E DAD12C 00740 JP C,BREAK ;Quit on Break
2C61 7E 00750 LD A,(HL) ;P/u the response
2C62 D631 00760 SUB '1' ;Adjust to binary
2C64 FE04 00770 CP 3+1 ;Be sure in range
2C66 30E7 00780 JR NC,LDOS3 ;Redo if not
00790 ;
00800 ; Convert drive address to select code
00810 ;
2C68 FE03 00820 CP 3 ;Convert 3 to 4
2C6A 3F 00830 CCF
2C6B CE00 00840 ADC A,0
2C6D FE01 00850 CP 1 ;Convert <0,1,2,4>
2C6F 17 00860 RLA ; to <1, 2, 4, 8>
2C70 47 00870 LD B,A ;Hang on to request
00880 ;
00890 ; Index the default drive code table
00900 ;
2C71 00910 LX805 EQU $
00920 IF @MOD2
00930 LD A,1 ;8"
00940 ENDIF
00950 IF @MOD4
2C71 3E00 00960 LD A,0 ;5"
00970 ENDIF
2C73 4F 00980 LD C,A
2C74 87 00990 ADD A,A ;Times 2
2C75 81 01000 ADD A,C ;Times 3
2C76 87 01010 ADD A,A ;Times 6
2C77 81 01020 ADD A,C ;Times 7

```

```

2C78 21232E 010300 LD HL,DRV TAB$ ;Index into 5" or 8"
2C7B 85 010400 ADD A,L ; default table
2C7C 6F 010500 LD L,A
2C7D 8C 010600 ADC A,H
2C7E 95 010700 SUB L
2C7F 67 010800 LD H,A
2C80 23 010900 INC HL
2C81 7E 011000 LD A,(HL) ;P/u default DCT+4
2C82 E6F0 011100 AND 0F0H ;Remove drive select
2C84 B0 011200 OR B ;Merge in new one
2C85 77 011300 LD (HL),A ;Update the DCT
2C86 2B 011400 DEC HL
2C87 010700 011500 LD BC,7 ;Init for 7-byte move
2C8A D1 011600 POP DE ;DE => DCT$
2C8B D5 011700 PUSH DE ;Save DCT$ pointer
2C8C 13 011800 INC DE
2C8D 13 011900 INC DE
2C8E 13 012000 INC DE ;Index to DCT+3
2C8F EDB0 012100 LDIR
2C91 D1 012200 POP DE
2C92 D5 012300 PUSH DE ;Save start again
2C93 21702F 012400 LD HL,BUFFER+70H ;Index the default vector
2C96 0E03 012500 LD C,3 ;Move in driver vector
2C98 EDB0 012600 LDIR
2C9A D1 012700 POP DE
012800 ;
012900 ; Compute the actual drive number used
013000 ;
2C9B 013100 @@GTDCT ;Get drive 0(ldir set C=0)
2C9B 3E51 000240 LD A,81
2C9D EF 000250 RST 40
2C9E FDE5 013200 PUSH IY ;Pass to HL for sub
2CA0 E1 013300 POP HL ;HL => start DCT's
2CA1 EB 013400 EX DE,HL ;DE=start, HL=current
2CA2 B7 013500 OR A ;Clear carry
2CA3 ED52 013600 SBC HL,DE ;HL = offset from start
2CA5 0E0A 013700 LD C,10 ;DCT length
2CA7 013800 @@DIV16 ;HL+A = HL/C
2CA7 3E5E 000260 LD A,94
2CA9 EF 000270 RST 40
2CAA 4D 013900 LD C,L ;Result = drive #
2CAB 014000 @@RSTOR ;Restore drive
2CAB 3E2C 000280 LD A,44
2CAD EF 000290 RST 40
2CAE 210000 014100 LD HL,0 ;Set no error return
2CB1 C9 014200 RET ;Init complete
014300 ;
014400 ; Routines to read/write the config sector
014500 ;
2CB2 21002F 014600 GETCFG LD HL,BUFFER ;Use buffer for I/O
014700 ;
014800 IF @MOD2
014900 LD C,L ;Pass drive #
015000 PUSH IY ;Save IY
2CB5 015100 @@GTDCT ;Fetch DCT
000300 LD A,81
000310 RST 40
015200 LD A,(IY+3) ;Get data
015300 AND 28H ;Bit 5/3
015400 CP 20H ;8" floppy?
015500 JR NZ,SETSYS1 ;Go if not

```

```

01560      LD      A,(IY+4)      ;Fetch data
01570      AND     50H          ;Bit 6/4
01580      CP      40H          ;DD not alien?
01590      JR      NZ,SETSYS1    ;Go if not
01600      LD      D,(IY+9)      ;Get dir cyl
01610      LD      E,L          ;Sector 0
2CB5      01620      @ORDSEC      ;Read sector
00032      LD      A,49
00033      RST     40
01630      CP      6            ;Directory?
01640      JR      NZ,SETSYS2    ;Nope, error
01650      LD      A,(BUFFER+0CDH) ;Get GAT data
01660      BIT     7,A          ;System disk?
01670 SETSYS1 LD      DE,0<8+2  ;Normal sysinfo sector
01680      JR      NZ,$+3        ;Go if data disk
01690      INC     D              ; else sysinfo on 1
01700      XOR     A              ;Set Z for no error
01710 SETSYS2 POP     IY        ;Restore DCT
01720      RET     NZ            ;Go if error
01730      ENDIF
01740 ;
01750      IF      @MOD4
2CB5 110200 01760      LD      DE,0<8+2      ;Get Config sector
01770      ENDIF
2CB8 4D      01780      LD      C,L          ; of system drive
2CB9      01790      @ORDSEC      ;Read it into core
2CB9 3E31    00034      LD      A,49
2CBB EF      00035      RST     40
2CBC C9      01800      RET
01810 ;
2CBD 6F      01820 IOERR LD      L,A          ;Error # to HL
2CBE 2600    01830      LD      H,0
2CC0 F6C0    01840      OR      0C0H          ;Abbrev, return
2CC2      01850      @ERROR      ;Display the error
2CC2 3E1A    00036      LD      A,26
2CC4 EF      00037      RST     40
2CC5      01860      @CKBRKC      ;Clear any Break
2CC5 3E6A    00038      LD      A,106
2CC7 EF      00039      RST     40
2CC8 C9      01870      RET
01880 ;
01890 ;      Internal error display routine
01900 ;
2CC9 216E2D 01910 VIASET LD      HL,VIASET$    ;"Install with SYSTEM
2CCC DD      01920      DB      0DDH
2CCD 218F2D 01930 ACTIVE LD      HL,ACTIVE$    ;"Drive in use
2CD0 DD      01940      DB      0DDH
2CD1 21CB2D 01950 BREAK  LD      HL,BREAK$     ;"Command aborted
2CD4 DD      01960      DB      0DDH
2CD5 21AD2D 01970 NODRV LD      HL,NODRV$     ;"Need a drive #
2CD8      01980      @LOGOT
00040      IFEQ     00H,1
00041      LD      HL,
00042      ENDIF
2CD8 3E0C    00043      LD      A,12
2CDA EF      00044      RST     40
2CDB 21FFFF 01990      LD      HL,-1          ;Set abort code
2CDE      02000      @CKBRKC      ;Clear any break
2CDE 3E6A    00045      LD      A,106
2CE0 EF      00046      RST     40
2CE1 C9      02010      RET

```

```

02020 ;
2CE2 0A 02030 HELLO$ DB LF,'FLOPPY Setup'
46 4C 4F 50 50 59 20 53
65 74 75 70
2CEF 02040 *GET CLIENT:3
03950 ;CLIENTS/ASM - File to establish sign-on headers
03960 ;
2CEF 20 03970 DB ' - 6.2.0 - Copyright 1982/83/84 by Logical'
2D 20 36 2E 32 2E 30 20
2D 20 43 6F 70 79 72 69
67 68 74 20 31 39 38 32
2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61
6C
2D19 20 03980 DB ' Systems, Inc. ',10
53 79 73 74 65 6D 73 2C
20 49 6E 63 2E 20 20 20
20 20 20 0A
03990 ;
2D2E 41 04000 DB 'All Rights Reserved. Licensed 1982/83/84'
6C 6C 20 52 69 67 68 74
73 20 52 65 73 65 72 76
65 64 2E 20 4C 69 63 65
6E 73 65 64 20 31 39 38
32 2F 38 33 2F 38 34
2D56 20 04010 DB ' to xxxxxxxxxxxxxxxxxxx',10,13
74 6F 20 78 78 78 78 78
78 78 78 78 78 78 78 78
78 78 78 78 78 0A 0D
02050 ;
2D6E 4D 02060 VIASET$ DB 'Must install via SYSTEM (DRIVER=',CR
75 73 74 20 69 6E 73 74
61 6C 6C 20 76 69 61 20
53 59 53 54 45 4D 20 28
44 52 49 56 45 52 3D 0D
2D8F 44 02070 ACTIVE$ DB 'Drive slot is already enabled',CR
72 69 76 65 20 73 6C 6F
74 20 69 73 20 61 6C 72
65 61 64 79 20 65 6E 61
62 6C 65 64 0D
2DAD 4C 02080 NODRV$ DB 'Logical drive number required',CR
6F 67 69 63 61 6C 20 64
72 69 76 65 20 6E 75 6D
62 65 72 20 72 65 71 75
69 72 65 64 0D
2DCB 43 02090 BREAK$ DB 'Command aborted',CR
6F 6D 6D 61 6E 64 20 61
62 6F 72 74 65 64 0D
2DDB 20 02100 DRVTP$ DB ' Enter drive code (0=5", 1=8") > ',3
20 20 45 6E 74 65 72 20
64 72 69 76 65 20 63 6F
64 65 20 28 30 3D 35 22
2C 20 31 3D 38 22 29 20
3E 20 03
2DFF 20 02110 DRVADR$ DB ' Enter drive I/O address <1-4> > ',3
20 20 45 6E 74 65 72 20
64 72 69 76 65 20 49 2F
4F 20 61 64 64 72 65 73
73 20 3C 31 2D 34 3E 20
3E 20 03
02120 DRVTP$

```

```

02130 ;
02140 ;          5" drive table
02150 ;
2E23 44      02160      DB      01000100B      ;5", 6ms, delay=n
2E24 40      02170      DB      01000000B      ;DDEN
2E25 FF      02180      DB      0FFH          ;Start cylinder
2E26 27      02190      DB      40-1          ;40 track drive
2E27 11      02200      DB      18-1          ;18 sec per cyl
2E28 45      02210      DB      3-1<5+6-1      ;6 sec/gran, 3 gran/cyl
2E29 14      02220      DB      40/2          ;Directory track
02230 ;
02240 ;          8" table
02250 ;
02260      IF      @MOD4
2E2A 21      02270      DB      00100001B      ;8", 6ms step
2E2B 40      02280      DB      01000000B      ;DDEN
2E2C FF      02290      DB      0FFH          ;Start cylinder
2E2D 4C      02300      DB      77-1          ;77 track drive
2E2E 0F      02310      DB      16-1          ;16 sec per cyl
2E2F 27      02320      DB      2-1<5+8-1      ;8 sec/gran, 2 gran/cyl
2E30 26      02330      DB      77/2          ;Directory track
02340      ENDIF
02350 ;
02360      IF      @MOD2
02370      DB      01100010B      ;+3 - 8", DD, 10ms, delay
02380      DB      01000000B      ;+4 - DDen capable
02390      DB      4CH          ;+5 - current cyl
02400      DB      77-1          ;+6 - high cylinder
02410      DB      0<5+29      ;+7 - sides + high sec
02420      DB      2<5+9      ;+8 - grans/cyl + sec/grn
02430      DB      77/2          ;+9 - dir cylinder
02440      ENDIF
02450 ;
0002      02460      BUF      DS      2
2F00      02470      ORG      $<-8+1<+8
0100      02480      BUFFER  DS      256
02490 ;
2C00      02500      END      BEGIN

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
ACTIVE	2CCD ACTIVE\$	2D8F BEGIN	2C00
BEGIN	2C09 BREAK	2CD1 BREAK\$	2DCB
BUF	2E31 BUFFER	2F00 CR	000D
DRVADR\$	2DFF DRV TAB\$	2E23 DRV TYP	2C33
DRV TYP\$	2DDB GETCFG	2CB2 HELLO\$	2CE2
IOERR	2CBD LDOS	2C11 LDOS3	2C4F
LF	000A LX805	2C71 NODRV	2CD5
NODRV\$	2DAD VIA SET	2CC9 VIA SET\$	2D6E
@@ABORT	81D9 @@ADTSK	826C @@BANK	8784
@@BKSP	8464 @@BREAK	879A @@CHNIO	81C4
@@CKBRKC	87E8 @@CKDRV	82C0 @@CKEOF	8479
@@CKTSK	8257 @@CLOSE	844F @@CLS	87D2
@@CMNDI	8203 @@CMNDR	8218 @@CTL	8028
@@DATE	819A @@DCSTAT	82FF @@DEBUG	8242
@@DECHEX	8704 @@DIRRD	8671 @@DIRWR	8686
@@DIV16	86EF @@DIV8	86DA @@DODIR	82D5
@@DSP	7FEC @@DSPLY	808C @@ERROR	822D
@@EXIT	81EE @@FEXT	85DE @@FLAGS	876E
@@FNAME	85F3 @@FSPEC	85C9 @@GATRD	865C
@@GATWR	869B @@GET	8000 @@GTDCB	861D
@@GTDCT	8608 @@GTMOD	8632 @@HDFMT	83A7
@@HEX16	8743 @@HEX8	872E @@HEXDEC	8719
@@HIGH\$	8758 @@INIT	8425 @@KBD	8064
@@KEY	7FD8 @@KEYIN	8078 @@KLTSK	82AB
@@LOAD	859F @@LOC	848E @@LOF	84A3
@@LOGGER	80C3 @@LOGOT	80D8 @@MSG	810F
@@MUL16	86C5 @@MUL8	86B0 @@OPEN	843A
@@PARAM	8185 @@PAUSE	8170 @@PEOF	84B8
@@PSN	84CD @@PRINT	8124 @@PRT	803C
@@PUT	8014 @@RAMDIR	82EA @@RDSEC	837D
@@RDSSC	8647 @@READ	84E2 @@REMOV	8410
@@RENAM	83FB @@REW	84F7 @@RMTSK	8281
@@RPTSK	8296 @@RREAD	850C @@RSLCT	8368
@@RSTOR	8329 @@RUN	85B4 @@RWRIT	8521
@@SEEK	8353 @@SEEKSC	8536 @@SKIP	854B
@@SLCT	8314 @@STEPI	833E @@TIME	81AF
@@VDCTL	815B @@VER	8560 @@VRSEC	8392
@@WEOF	8575 @@WHERE	8050 @@WRITE	858A
@@WRSEC	83BC @@WRSSC	83D1 @@WRTRK	83E6

2C00 is the transfer address

00000 Total errors

NOTES:

NOTES:

FORMAT/CMD - Disk initialization program

The Format utility allows a floppy or hard disk to be initialized. Parameters are available to set the number of sides and cylinders, select the density, and set a boot step rate for system disks.

```

00100 ;FORMAT1/ASM - Format Program
0000 00110 TITLE <FORMAT - LS-DOS 6.2>
0000 00120 SUBTTL '<Format Execution Code>'
00130 ;
42E0 00140 PASSWORD EQU 42E0H
0062 00150 RLS EQU 62H
000A 00160 LF EQU 10
000D 00170 CR EQU 13
3C00 00180 CRT3 EQU 3C00H
F800 00190 CRT4 EQU 0F800H
00200 ;
0000 00210 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00220 *GET COPYCOM:3 ;Copyright message
03920 ; COPYCOM - File for Copyright COMMENT block
03930 ;
0000 03940 COM '<*(C) 1982,83,84 by LSI*>'
00230 *LIST ON
00240 ;
2600 00250 ORG 2600H
00260 ;
00270 IF @MOD4
2600 9D 00280 BOOTST$ DB 9DH ;Boot step rate offset
00290 ENDIF
00300 IF @MOD2
00310 BOOTST$ DB 03H
00320 ENDIF
00330 ;
2601 FDCB0466 00340 GOFMT BIT 4,(IY+4) ;Jump if alien controller
2605 C2C127 00350 JP NZ,HRDRV
2608 110000 00360 FMTTBL LD DE,0 ;P/u table pointer
260B 1A 00370 LD A,(DE) ;P/u # of sectors to fmt
260C 13 00380 INC DE ;Adj for zero offset
260D 32E92A 00390 LD (SECTRK),A
2610 47 00400 LD B,A
2611 FDCB046E 00410 BIT 5,(IY+4) ;Need twice as many
2615 2801 00420 JR Z,$+3 ; if 2-sided drive
2617 07 00430 RLCA
2618 32E82A 00440 LD (SECCYL),A
261B 210000 00450 SYSPRM LD HL,0 ;P/u system info parm
261E 7C 00460 LD A,H ;Don't format if system
261F B5 00470 OR L ; info only is req
2620 C29427 00480 JP NZ,MOVFREE
2623 1A 00490 LD A,(DE) ;P/u track skew
2624 13 00500 INC DE
2625 320327 00510 LD (TRKSKEW+1),A
2628 ED53A226 00520 LD (SECSKEW+1),DE ;Format sector skew
00530 ;
00540 ; Index past sector info
00550 ;
262C 3C 00560 INC A ;Add DE -> begin of sec #
262D 80 00570 ADD A,B ;B -> # of sectors/side
262E 83 00580 ADD A,E ; A+1 -> a code byte
262F 5F 00590 LD E,A
2630 8A 00600 ADC A,D
2631 93 00610 SUB E
2632 57 00620 LD D,A
2633 210031 00630 LD HL,FORMAT ;Buffer for format data
2636 010030 00640 LD BC,HITBUF ;Tempy ptrs to trk,sect info

```

Format Execution Code

```

00650 ;
00660 ;      Create the formatting data without trk,sect info
00670 ;
2639 1A 00680 FMTDAT LD      A,(DE)      ;P/u table format byte
263A 13 00690      INC      DE           ;Bump table ptr
263B FEF1 00700      CP      0F1H        ;Start of cylinder?
263D 282A 00710      JR      Z,CODF1
263F FEF2 00720      CP      0F2H        ;Start of track trailer?
2641 282D 00730      JR      Z,CODF2
2643 FEF3 00740      CP      0F3H        ;Start of track ID info?
2645 2833 00750      JR      Z,CODF3
2647 FEF4 00760      CP      0F4H        ;End of table parms?
2649 2837 00770      JR      Z,CODF4
264B FEF5 00780      CP      0F5H        ;Start of data?
264D C5 00790      PUSH     BC
264E 200F 00800      JR      NZ,CODE1     ;Go if not
00810 ;
00820 ;      Write 2 byte data pattern to format buffer
00830 ;
2650 1A 00840      LD      A,(DE)      ;P/u length to write
2651 13 00850      INC      DE           ;Bump to 1st data byte
2652 47 00860      LD      B,A         ;Xfer length to B
2653 1A 00870      LD      A,(DE)      ;P/u a data byte
2654 13 00880      INC      DE           ;Bump again for 2nd byte
2655 4F 00890      LD      C,A         ;Xfer 1st byte
2656 1A 00900      LD      A,(DE)      ;P/u 2nd byte
2657 71 00910 CODF5 LD      (HL),C     ;Stuff into buf
2658 23 00920      INC      HL
2659 77 00930      LD      (HL),A
265A 23 00940      INC      HL
265B 10FA 00950      DJNZ    CODF5      ;Loop til xfered
265D 1806 00960      JR      CODRET
00970 ;
00980 ;      Xfer bytes to the format buffer area
00990 ;      A => count to move
01000 ;      DE => data byte to duplicate
01010 ;
265F 47 01020 CODE1 LD      B,A         ;Count to B
2660 1A 01030      LD      A,(DE)      ;P/u data byte to move
2661 77 01040 CODE1A LD      (HL),A    ;Fill buf with byte
2662 23 01050      INC      HL
2663 10FC 01060      DJNZ    CODE1A     ;Loop til done
2665 C1 01070 CODRET POP      BC
2666 13 01080      INC      DE           ;Bump table ptr
2667 18D0 01090      JR      FMTDAT     ;Back for more
01100 ;
01110 ;      Save the current table posn and the number of
01120 ;      sectors per cylinder on the stack.
01130 ;
2669 3AE92A 01140 CODF1 LD      A,(SECTRK) ;P/u # of sectors/side
266C D5 01150 CODF1A PUSH     DE         ;Save table pointer
266D F5 01160      PUSH     AF         ;Save value
266E 18C9 01170      JR      FMTDAT
01180 ;
01190 ;      Done with a sector. Are there more on this cyl?
01200 ;
2670 F1 01210 CODF2 POP      AF         ;Count down the # of
2671 3D 01220      DEC      A           ; sectors to format
2672 2803 01230      JR      Z,CODF2A   ;Go if last one done

```

Format Execution Code

```

2674 D1      01240      POP      DE          ;Recover table ptr
2675 18F5     01250      JR        CODF1A      ;Loop for more
                01260 ;
2677 F1      01270 CODF 2A POP      AF          ;Clean the stack
2678 18BF     01280      JR        FMTDAT      ; and finish off the cyl
                01290 ;
                01300 ;      Build a table of the location in the format buffer of
                01310 ;      the track and sector ID bytes, to be filled in during
                01320 ;      the actual formatting.
                01330 ;
267A 7D      01340 CODF 3  LD        A,L          ;Stuff pointer to where
267B 02      01350      LD        (BC),A        ; track & sector info
267C 03      01360      INC        BC          ; is to be placed
267D 7C      01370      LD        A,H
267E 02      01380      LD        (BC),A
267F 03      01390      INC        BC
2680 18B7     01400      JR        FMTDAT
                01410 ;
                01420 ;      Finished building format cyl info. Terminate the ID table
                01430 ;      with an extra 256 bytes in case of overrun.
                01440 ;
2682 ED536527 01450 CODF 4  LD        (VERSKW+1),DE ;Table posn of verify order
2686 AF      01460      XOR        A          ;Stuff two X'00's to
2687 02      01470      LD        (BC),A        ; indicate the end
2688 03      01480      INC        BC          ; of the ID posn table
2689 02      01490      LD        (BC),A
268A 0600     01500      LD        B,0          ;Stuff 256 FF's into the
268C 3EFF     01510      LD        A,0FFH      ; format buffer
268E 77      01520      LD        (HL),A
268F 23      01530      INC        HL
2690 10FC     01540      DJNZ      $-2
                01550 ;
                01560 ;      Begin the formatting
                01570 ;
2692         01580      @@DSPLY FMTCYL$        ;"formatting clinder...
                00001      IFEQ      01H,1
2692 21392C   00002      LD        HL,FMTCYL$
                00003      ENDIF
2695 3E0A     00004      LD        A,10
2697 EF      00005      RST        40
2698 FD7E05   01590 BGNFMT LD        A,(IY+5)    ;P/u cylinder position
269B CD502A   01600      CALL      CVDEC        ;Cvrt to decimal
269E CD892A   01610      CALL      DSPCYL
26A1 010000   01620 SECSKEW LD        BC,0      ;Begin of sector table
26A4 210030   01630 BFMT1  LD        HL,HITBUF  ;P/u ptr to ID posn table
                01640 ;
                01650 BFMT2
26A7         01660      @@CKBRKC        ;Check for break
26A7 3E6A     00006      LD        A,106
26A9 EF      00007      RST        40
26AA C2B929   01670      JP        NZ,BREAK    ;Go if so
                01680 ;
26AD 5E      01690      LD        E,(HL)      ;P/u positions having
26AE 23      01700      INC        HL          ; sector & cylinder
26AF 56      01710      LD        D,(HL)      ; info to be stuffed
26B0 23      01720      INC        HL          ; into format data
26B1 7A      01730      LD        A,D        ;Finished?
26B2 B3      01740      OR        E
26B3 2820     01750      JR        Z,BFMT4

```

Format Execution Code

```

26B5 FD7E05 01760 LD A,(IY+5) ;P/u cylinder # & stuff
26B8 12 01770 LD (DE),A ; into format data
26B9 13 01780 INC DE
26BA FD7E03 01790 LD A,(IY+3) ;Stuff the side-select
26BD E610 01800 AND 10H ; bit
26BF 0F 01810 RRCA
26C0 0F 01820 RRCA
26C1 0F 01830 RRCA
26C2 0F 01840 RRCA
26C3 12 01850 LD (DE),A ; into the format data
26C4 13 01860 INC DE
26C5 0A 01870 LD A,(BC) ;P/u the sector number
26C6 B7 01880 OR A
26C7 F2CF26 01890 JP P,BFMT3 ;Go if a good number
26CA 81 01900 ADD A,C ; else off the end,
26CB 4F 01910 LD C,A ; calculate the beginning
26CC 3801 01920 JR C,BFMT3 ; of the sector table
26CE 05 01930 DEC B
26CF 0A 01940 BFMT3 LD A,(BC) ;P/u the next sector #
26D0 12 01950 LD (DE),A ; and stuff in format data
26D1 13 01960 INC DE
26D2 03 01970 INC BC
26D3 18D2 01980 JR BFMT2 ;Loop until cylinder done
01990 ;
26D5 ED43A226 02000 BFMT4 LD (SECSKEW+1),BC ;Save end of sector table
26D9 FD5605 02010 LD D,(IY+5) ;P/u current cylinder
26DC 210031 02020 LD HL,FORMAT ;Pt to format data
26DF CDF A29 02030 CALL SELECT ;Drive select
26E2 C2A529 02040 JP NZ,IOERR ;Go on error
26E5 CD0E2A 02050 CALL WRCYL ;Cylinder write
26E8 C2A529 02060 JP NZ,IOERR
26EB FDCB046E 02070 BIT 5,(IY+4) ;Double sided?
26EF 280D 02080 JR Z,BFMT5
26F1 FDCB0366 02090 BIT 4,(IY+3) ;Flip bit for 2nd side
26F5 2007 02100 JR NZ,BFMT5 ; if not already on it,
26F7 FDCB03E6 02110 SET 4,(IY+3) ; else go to next
26FB 03 02120 INC BC ;Bump to start side 2
26FC 18A6 02130 JR BFMT1 ; at different sector #
26FE FDCB03A6 02140 BFMT5 RES 4,(IY+3) ;Turn off side 2
2702 3E00 02150 TRKSKEW LD A,0 ;P/u the track skew byte
2704 81 02160 ADD A,C ;Repoint to beginning
2705 4F 02170 LD C,A ; of sector table
2706 88 02180 ADC A,B ;Skew start of next track
2707 91 02190 SUB C
2708 47 02200 LD B,A
2709 ED43A226 02210 LD (SECSKEW+1),BC
270D FD7E05 02220 LD A,(IY+5) ;Finished?
2710 FDBE06 02230 CP (IY+6)
2713 2820 02240 JR Z,BGNVER ;Begin verify if so
2715 014200 02250 LD BC,1000/15 ;Approx 1 ms pause
2718 02260 @@PAUSE ; before STEPIN
2718 3E10 00008 LD A,16
271A EF 00009 RST 40
271B CD042A 02270 CALL STEPIN ;Step in
271E C2A529 02280 JP NZ,IOERR ;Go on error
2721 019826 02290 LD BC,BGNFMT ;Place RET addr on stack
2724 CD092A 02300 CKWAIT CALL RSELECT ;Wait for idle FDC
2727 C2A529 02310 JP NZ,IOERR ;Go on error
272A C5 02320 PUSH BC ;Save RET addr

```

Format Execution Code

```

02330 ;
02340 ;      WAIT parameter for time delay after STEPIN
02350 ;
272B 01C800 02360 WAITPRM LD      BC,3000/15      ;Approx 3 ms delay
272E 78      02370      LD      A,B              ; after STEPIN
272F B1      02380      OR      C
2730 C8      02390      RET      Z              ;Do next track if no wait
2731         02400      @@PAUSE              ; else wait for count
2731 3E10    00010      LD      A,16
2733 EF      00011      RST      40
2734 C9      02410      RET
02420 ;
02430 ;      Begin the verification process
02440 ;
2735 0E0D    02450 BGNVER LD      C,CR              ;Posn to next dsply line
2737         02460      @@DSP
2737 3E02    00012      LD      A,2
2739 EF      00013      RST      40
273A CDFF29 02470      CALL     RESTOR              ;Restore to cyl 0
273D 206A    02480      JR      NZ,BVER9            ;Go on error
273F         02490      @@DSPLY VERCYL$            ;"verifying cylinder..."
00014      IFEQ      01H,1
273F 21512C 00015      LD      HL,VERCYL$
00016      ENDIF
2742 3E0A    00017      LD      A,10
2744 EF      00018      RST      40
2745 1600    02500      LD      D,0              ;Init track count
02510 BVER1
2747         02520      @@CKBRKC              ;Check for break
2747 3E6A    00019      LD      A,106
2749 EF      00020      RST      40
274A C2B929 02530      JP      NZ,BREAK            ; and abort if so
02540 ;
274D 6A      02550      LD      L,D              ;Pt to GAT byte for this
274E 262E    02560      LD      H,GATBUF<-8        ; track & bypass verify
2750 7E      02570      LD      A,(HL)            ; if track not formatted
2751 3C      02580      INC      A
2752 2836    02590      JR      Z,BVER8
02600 ;
2754 7A      02610      LD      A,D
2755 CD502A   02620      CALL     CVDEC              ;Convert cyl # to ASCII
2758 D5      02630      PUSH     DE
2759 CD892A   02640      CALL     DSPCYL            ;Display the current cyl
275C D1      02650      POP      DE
275D AF      02660      XOR      A              ;Initialize starting sector
275E 327227 02670      LD      (BVER5+1),A
2761 326927 02680      LD      (BVER4+1),A
2764 010000 02690 VERSKEW LD      BC,0              ;P/u start of sector tbl
2767 0A      02700 BVER3 LD      A,(BC)            ;P/u sector #
2768 C600    02710 BVER4 ADD      A,0              ;Add in a side's sectors
276A 5F      02720      LD      E,A              ; if on side 2
276B CD272A 02730      CALL     VERSEC            ;Sector verify
276E 2039    02740      JR      NZ,BVER9            ;Go on error
2770 03      02750      INC      BC              ;Bump sector table ptr
2771 3E00    02760 BVER5 LD      A,0              ;P/u sector #
2773 3C      02770      INC      A              ;Bump it up
2774 327227 02780      LD      (BVER5+1),A        ; and save new #
2777 5F      02790      LD      E,A              ;Xfer to sector register
2778 3AE82A 02800      LD      A,(SECCYL)          ;Is this = a cyl?

```

Format Execution Code

```

277B BB      02810 CP      E
277C 280C    02820 JR      Z,BVER8      ;Go if cyl done
277E 3AE92A  02830 LD      A,(SECTRK)    ;Is this a track's worth?
2781 BB      02840 CP      E
2782 20E3    02850 JR      NZ,BVER3     ;Loop if not
2784 326927  02860 LD      (BVER4+1),A    ;Update the add for side2
2787 03      02870 INC      BC
2788 18DA    02880 JR      VERSKEW
          02890 ;
          02900 ;      Readjust for end of cylinder
          02910 ;
278A 7A      02920 BVER8 LD      A,D      ;P/u current cyl position
278B 14      02930 INC      D      ;Bump to next cyl
278C FDBE06  02940 CP      (IY+6)      ;Cp to highest # cyl
278F 014727  02950 LD      BC,BVER1     ;Go if more to verify
2792 2090    02960 JR      NZ,CKWAIT    ; after checking WAIT
          02970 ;
          02980 ;      Shift the FREE table to LOCKOUT table
          02990 ;
2794 21002E  03000 MOVFREE LD      HL,GATBUF    ;Ptr to allocation info
2797 11602E  03010 LD      DE,GATBUF+60H    ;Lockout table
279A 0600    03020 LD      B,0
279C FD4E06  03030 LD      C,(IY+6)      ;P/u hi cyl
279F 0C      03040 INC      C      ;Offset from 0
27A0 EDB0    03050 LDIR             ;Shift info to the lockout tbl
27A2 0E0D    03060 LD      C,CR      ;Print a newline
27A4         03070 @@DSP
27A4 3E02    00021 LD      A,2
27A6 EF      00022 RST      40
27A7 185D    03080 JR      CALCDIR      ;Go finish DIR init
          03090 ;
          03100 ;      Got verify error
          03110 ;
27A9 FE05    03120 BVER9 CP      5      ;Data rec not found?
27AB 2805    03130 JR      Z,BVER10
27AD FE04    03140 CP      4      ;Parity error?
27AF C2A529  03150 JP      NZ,IOERR      ;Quit on any other
27B2 D5      03160 BVER10 PUSH     DE
27B3         03170 @@DSPLY STAR$      ;Show the * lockout
          00023 IFEQ     01H,1
27B3 21692C  00024 LD      HL,STAR$
          00025 ENDIF
27B6 3E0A    00026 LD      A,10
27B8 EF      00027 RST      40
27B9 D1      03180 POP      DE
27BA 6A      03190 LD      L,D      ;Pt to this cyl
27BB 262E    03200 LD      H,GATBUF<-8    ; in the GAT
27BD 36FF    03210 LD      (HL),0FFH    ;Lockout this cylinder
27BF 18C9    03220 JR      BVER8      ;Continue verifying
          03230 ;
          03240 ;      Hard drive format - most work done by controller
          03250 ;
27C1 218C39  03260 HRDRV LD      HL,LASTMSG    ;Give one last chance to
27C4 FDCB035E 03270 BIT      3,(IY+3)    ; abort before wiping
27C8 2809    03280 JR      Z,AFLOP      ; disk unless floppy
27CA CD5D2A  03290 CALL     GET3      ;Is hard, get response
27CD 7E      03300 LD      A,(HL)      ;P/u 1st char of resp
27CE FE59    03310 CP      'Y'      ;Must be yes to continue
27D0 C2B929  03320 JP      NZ,FMTABT

```


Format Execution Code

```

27D3 3A1C26 03330 AFLOP LD A,(SYSPRM+1) ;Bypass the formatting
27D6 B7 03340 OR A ; if system info only
27D7 200C 03350 JR NZ,HRDRV1
27D9 03360 @@DSPLY FMTG$ ;"formatting - be patient
00028 IFEQ 01H,1
27D9 216E2C 00029 LD HL,FMTG$
00030 ENDIF
27DC 3E0A 00031 LD A,10
27DE EF 00032 RST 40
27DF CD132A 03370 CALL FMTHD ;Format hard drive
27E2 C2A529 03380 JP NZ,IOERR
27E5 FD7E07 03390 HRDRV1 LD A,(IY+7) ;# of sectors/gran
27E8 57 03400 LD D,A ;-> reg E
27E9 E61F 03410 AND 1FH
27EB 5F 03420 LD E,A
27EC 1C 03430 INC E ;Bump for 0 offset
27ED AA 03440 XOR D
27EE 07 03450 RLCA ;Get # of heads
27EF 07 03460 RLCA ;Into reg D
27F0 07 03470 RLCA
27F1 3C 03480 INC A ;Adjust for zero offset
27F2 4F 03490 LD C,A
27F3 03500 @@MUL8 ;Multiply E x C
27F3 3E5A 00033 LD A,90
27F5 EF 00034 RST 40
27F6 FDCB046E 03510 BIT 5,(IY+4) ;2-sided?
27FA 2801 03520 JR Z,$+3
27FC 87 03530 ADD A,A ;Twice the number
27FD 32E82A 03540 LD (SECCYL),A
2800 FDCB035E 03550 BIT 3,(IY+3) ;Floppy?
2804 288E 03560 JR Z,MOVFREE ;Form lock table instead
03570 ;
03580 ; Routine to calculate the directory cylinder
03590 ;
2806 CDFF29 03600 CALCDIR CALL RESTOR ;Step in
2809 C2A529 03610 JP NZ,IOERR ;Go on error
280C 262E 03620 LD H,GATBUF<-8
280E FD6E06 03630 LD L,(IY+6) ;P/u highest # cylinder
2811 010000 03640 DIRPARM LD BC,0000 ;P/U 'DIR=' parm
2814 79 03650 LD A,C ;Check if entered
2815 B0 03660 OR B
2816 2806 03670 JR Z,NODIR ;Calc one if not entered
2818 BD 03680 CP L ;Entered so check if
2819 3003 03690 JR NC,NODIR ; within cylinders
281B 6F 03700 LD L,A ;Is ok, use it
281C 1803 03710 JR DIRSET
281E 2C 03720 NODIR INC L ;Adj for zero offset
281F CB3D 03730 SRL L ;Divide by 2 to find
2821 0E00 03740 DIRSET LD C,0 ; disk midpoint
03750 ;
03760 ; Perform expanding binary search to find
03770 ; A cylinder available for the directory
03780 ;
2823 7E 03790 CALC1 LD A,(HL) ;Is this cylinder
2824 3C 03800 INC A ;Available or locked out?
2825 2019 03810 JR NZ,GENSYS ;Bypass if available
2827 0C 03820 INC C ;Bump C
2828 79 03830 LD A,C
2829 0F 03840 RRCA ;Test if odd or even

```

Format Execution Code

```

282A 7D      03850      LD      A,L          ;Get current test pos
282B 3009    03860      JR      NC,CALC2      ;Jump if C was even
282D 81      03870      ADD     A,C          ;Add to previous pos
282E 6F      03880      LD      L,A
282F FDBE06  03890      CP      (IY+6)        ;Go over the top?
2832 20EF    03900      JR      NZ,CALC1      ;Loop if not
2834 1804    03910      JR      CALC3          ;Else abort
2836 91      03920      SUB     C            ;Try a lower cylinder #
2837 6F      03930      LD      L,A
2838 20E9    03940      JR      NZ,CALC1      ;At cylinder 0?
283A 217C2C  03950      LD      HL,NOCYL$      ;"no dir space avail...
283D C3B929  03960      JP      FMTABT
                03970      ;
                03980      ;      Generate the system initialization
                03990      ;
2840 FD7509  04000      GENSYS LD      (IY+9),L      ;Stuff the dir cyl
2843 7D      04010      LD      A,L
2844 CD502A  04020      CALL   CVDEC          ;Cvrt reg A to 2 dec digs
2847 ED43C62C 04030      LD      (DIRASC$),BC      ;Stuff into the message
284B        04040      @@DSPLY DIRCYL$      ;"dir will be placed...
                04035      IFEQ     01H,1
284B 21A12C  04036      LD      HL,DIRCYL$
                04037      ENDIF
284E 3E0A    04038      LD      A,10
2850 EF      04039      RST      40
2851        04050      @@DSPLY IPLSYS$      ;"initializing...
                04040      IFEQ     01H,1
2851 21C92C  04041      LD      HL,IPLSYS$
                04042      ENDIF
2854 3E0A    04043      LD      A,10
2856 EF      04044      RST      40
2857 21002E  04060      LD      HL,GATBUF
285A 7E      04070      LD      A,(HL)          ;P/u GAT byte for 1st
285B F601    04080      OR      1            ; cylinder & show 1st
285D 77      04090      LD      (HL),A          ; gran in use for BOOTs
285E FD7E09  04100      LD      A,(IY+9)        ;Dir cyl # into DIR/SYS
2861 32D32A  04110      LD      (DIRDIR+16H),A
2864 6F      04120      LD      L,A            ;Show entire directory
2865 36FF    04130      LD      (HL),0FFH      ; cylinder used
                04140      ;
                04150      ;      Update BOOT for DIR & step rate
                04160      ;
2867 FD7E09  04170      LD      A,(IY+9)        ;Dir cyl into BOOT
286A 32022F  04180      LD      (BOOT+2),A
286D 3A0026  04190      LD      A,(BOOTST$)      ;P/u offset
2870 6F      04200      LD      L,A
2871 262F    04210      LD      H,BOOT<-8
2873 3AE72A  04220      LD      A,(STEPDFT)      ;P/u boot step rate
                04230      IF      @MOD2
                04240      OR      80H          ;Create single byte opcode
                04250      ENDIF
2876 77      04260      LD      (HL),A          ; & set into BOOT
2877 110000  04270      LD      DE,0          ;Init for cyl 0, sect 0
287A CD272A  04280      CALL   VERSEC          ;Test if formatted
287D 21352D  04290      LD      HL,NOTFMT$      ;"Can't, not formatted
2880 C2BC29  04300      JP      NZ,EXTERR      ;Error if not
2883 21002F  04310      LD      HL,BOOT          ;Pt to Data disk BOOT
2886 CD182A  04320      CALL   WRSEC          ; & write it
2889 CC442A  04330      CALL   Z,WRDIR1      ;Verify after write

```

Format Execution Code

```

288C C2A529 04340 JP NZ,IOERR ; & display '.'
288F 110100 04350 LD DE,1 ;Pt to cyl 0, sector 1
2892 21002F 04360 LD HL,BOOT ;Pt to the sector 1 boot
2895 CD182A 04370 CALL WRSEC ;Write 0/1
2898 CC442A 04380 CALL Z,WRDIR+3 ;Verify after write
289B C2A529 04390 JP NZ,IOERR
04400 ;
04410 ; Complete GAT construction
04420 ;
289E FD7E06 04430 LD A,(IY+6) ;P/u highest # cylinder
28A1 D622 04440 SUB 22H ; & adj offset from 34
28A3 32CC2E 04450 LD (GATBUF+0CCH),A ;Stuff GAT cyl excess
28A6 FD7E04 04460 LD A,(IY+4) ;P/u # of sides
28A9 E6A0 04470 AND 80H+20H
28AB 47 04480 LD B,A ;Save tempy in B
28AC FD7E03 04490 LD A,(IY+3) ;P/u density
28AF E640 04500 AND 40H ;Mask it,
28B1 B0 04510 OR B ; merge in sides
28B2 47 04520 LD B,A ; and save it
28B3 FD7E08 04530 LD A,(IY+8) ;P/u # of grans/cyl
28B6 07 04540 RLCA
28B7 07 04550 RLCA ; to bits 0-2
28B8 07 04560 RLCA
28B9 E607 04570 AND 7 ;Mask it
28BB 325229 04580 LD (CYLGRN+1),A
28BE B0 04590 OR B ;Merge the two
28BF F680 04600 OR 80H ;Show it's a data disk
28C1 32CD2E 04610 LD (GATBUF+0CDH),A ;Stuff into GAT
04620 ;
28C4 11F52E 04630 LD DE,GATBUF+255-10 ;6.2 Media Data Block
28C7 21DB28 04640 LD HL,LSIID ;Point to header
28CA 010400 04650 LD BC,04 ;Set length &
28CD EDB0 04660 LDIR ; move it
28CF FDE5 04670 PUSH IY ;Get DCT address
28D1 E1 04680 POP HL ; into HL
28D2 23 04690 INC HL ;Bypass the driver vector
28D3 23 04700 INC HL
28D4 23 04710 INC HL
28D5 0E07 04720 LD C,7 ;Bytes to move
28D7 EDB0 04730 LDIR
28D9 1804 04740 JR WRGAT1 ;Skip around string
28DB 03 04750 LSIID DB 03,'LSI'
4C 53 49
04760 ;
04770 ; Write copy of GAT into 0/3
04780 ;
04790 WRGAT1
28DF 21002E 04800 LD HL,GATBUF ;Pt to GAT buffer
28E2 1600 04810 LD D,0 ;Write it out to
28E4 1E03 04820 LD E,3 ;Cyl 0, sector 3
28E6 CD182A 04830 CALL WRSEC ;Write 0/3
28E9 CC442A 04840 CALL Z,WRDIR1 ;Verify after write
28EC C2A529 04850 JP NZ,IOERR ;Quit on error
04860 ;
04870 ; Write the system information sector
04880 ;
28EF 210030 04890 LD HL,HITBUF ;Zero out buffer
28F2 3600 04900 GSYS1 LD (HL),0
28F4 2C 04910 INC L

```

Format Execution Code

```

28F5 20FB      04920      JR      NZ,GSYS1
28F7 210030    04930      LD      HL,HITBUF      ;Set first byte to OSVER
28FA 3662      04940      LD      (HL),RLS      ; for release number
28FC 2E20      04950      LD      L,20H        ;Point h1 to AUTO buffer
28FE 360D      04960      LD      (HL),0DH      ;Put in terminator
2900 110200    04970      LD      DE,2         ;Pt to cyl 0, sector 2
2903 6A        04980      LD      L,D         ;H1 now points to HITBUF
2904 CD182A    04990      CALL     WRSEC       ;Write 0/2
2907 CC442A    05000      CALL     Z,WRDIR1     ;Verify after write
290A C2A529    05010      JP      NZ,IOERR     ;Quit on error
290D 2E20      05020      LD      L,20H        ;Zero this out for use
290F 3600      05030      LD      (HL),0      ; when writing HIT
                05040      ;
                05050      ;
                05060      ;
                Write out the directory GAT
2911 21002E    05070      LD      HL,GATBUF     ;Pt to GAT sector buffer
2914 FD5609    05080      LD      D,(IY+9)     ;P/u the dir cyl
2917 5D        05090      LD      E,L         ;Denote sector 0
2918 CD412A    05100      CALL     WRDIR     ;Write the GAT
291B C2A529    05110      JP      NZ,IOERR
                05120      ;
                05130      ;
                05140      ;
                Construct the HIT
291E 210030    05150      LD      HL,HITBUF     ;Point to the HIT buffer
2921 36A2      05160      LD      (HL),0A2H     ;Stuff BOOT/SYS hash code
2923 23        05170      INC      HL
2924 36C4      05180      LD      (HL),0C4H     ;Stuff DIR/SYS hash code
2926 2B        05190      DEC      HL
2927 FD5609    05200      LD      D,(IY+9)     ;P/u dir cyl #
292A 1E01      05210      LD      E,1         ;Pt to sector 1
292C CD412A    05220      CALL     WRDIR     ;Write the HIT
292F C2A529    05230      JP      NZ,IOERR
2932 110030    05240      LD      DE,HITBUF     ;Establish buffer for
2935 219D2A    05250      LD      HL,BOOTDIR    ; dir records
2938 012000    05260      LD      BC,32        ;Move BOOT/SYS dir record
293B EDB0      05270      LDIR             ; into 1st slot
293D FD5609    05280      LD      D,(IY+9)     ;P/u dir cyl
2940 1E02      05290      LD      E,2         ;This will be sector 2
2942 210030    05300      LD      HL,HITBUF     ;Pt to buffer start
2945 CD412A    05310      CALL     WRDIR     ;Write the sector
2948 C2A529    05320      JP      NZ,IOERR
294B 3AE82A    05330      LD      A,(SECCYL)    ;P/u # of records
294E 32D12A    05340      LD      (DIRDIR+14H),A ; & stuff into DIR/SYS
2951 3E00      05350      LD      A,0         ;P/u # grans/cyl
2953 FDCB046E  05360      BIT      5,(IY+4)     ;Test 2-sided
2957 2802      05370      JR      Z,$+4
2959 87        05380      ADD      A,A         ;Double count on 2-sided
295A 3C        05390      INC      A         ;Plus 1 for 0 offset adj
295B 32D42A    05400      LD      (DIRDIR+17H),A ;Stuf in DIR/SYS
295E FD7E09    05410      LD      A,(IY+9)     ;P/u the dir cyl # &
2961 32D32A    05420      LD      (DIRDIR+16H),A ; stuff into the DIR rec
2964 21BD2A    05430      LD      HL,DIRDIR    ;Pt to start of DIR data
2967 110030    05440      LD      DE,HITBUF     ;Pt to start of dir buf
296A 012000    05450      LD      BC,32        ;Move DIR/SYS into buf
296D EDB0      05460      LDIR
296F FD5609    05470      LD      D,(IY+9)     ;P/u dir cyl #
2972 1E03      05480      LD      E,3         ;Write as sector 3
2974 210030    05490      LD      HL,HITBUF     ;Pt to start of buffer
2977 CD412A    05500      CALL     WRDIR     ;Write the sector

```

Format Execution Code

```

297A 2029      05510      JR      NZ,IOERR
297C 210030    05520      LD      HL,HITBUF      ;Zero the 1st 32 bytes
297F 0620      05530      LD      B,32          ; of the buffer to clear
2981 3600      05540      LD      (HL),0        ;Where we stuffed the
2983 23        05550      INC      HL           ; BOOT & DIR dir records
2984 10FB      05560      DJNZ     GSYS2
2986 FD5609    05570      LD      D,(IY+9)      ;P/u dir cyl #
2989 1E04      05580      LD      E,4          ;Cont writing at sect 4
298B 210030    05590      LD      HL,HITBUF      ;Pt to start of buffer
298E CD412A    05600      CALL     WRDIR         ;Write the sector
2991 2012      05610      JR      NZ,IOERR
                05620 ;
                05630 ;      Write the remaining directory
                05640 ;
2993 1C        05650      INC      E           ;Bump the sector pointer
2994 3AE82A    05660      LD      A,(SECCYL)      ;P/u highest # sector
2997 BB        05670      CP      E           ;Are we finished yet?
2998 20F1      05680      JR      NZ,GSYS3      ;Loop if not
299A CDDF29    05690      CALL     EXIT2         ;Get system disk
299D           05700      @@DSPLY FMTCA0$      ;"formatting complete..."
                00045      IFEQ     01H,1
299D 210E2D    00046      LD      HL,FMTCA0$
                00047      ENDIF
29A0 3E0A      00048      LD      A,10
29A2 EF        00049      RST      40
29A3 1823      05710      JR      EXIT
                05720 ;
                05730 ;      Exit procedures
                05740 ;
29A5 F5        05750      IOERR  PUSH     AF           ;Save errcod
29A6 CDDF29    05760      CALL     EXIT2         ;Interrupts on if needed
29A9 F1        05770      POP      AF           ;Rcvr errcod
29AA FE3F      05780      CP      63          ;Extended error?
29AC 280E      05790      JR      Z,EXTERR      ;Go if so
29AE 6F        05800      LD      L,A          ;Error code to HL
29AF 2600      05810      LD      H,0
29B1 F6C0      05820      OR      0C0H         ;Mask to ABORT with brief
29B3 4F        05830      LD      C,A          ;Error code to C
29B4           05840      @@ERROR      ; for error display
29B4 3E1A      00050      LD      A,26
29B6 EF        00051      RST      40
29B7 180C      05850      JR      ERREXIT
                05860 ;
29B9           05870      BREAK  EQU      $
29B9 21242D    05880      FMTABT LD      HL,FMTABT$      ;"Command aborted
29BC           05890      EXTERR @@LOGOT      ;Some error to abort job
                00052      IFEQ     00H,1
                00053      LD      HL,
                00054      ENDIF
29BC 3E0C      00055      LD      A,12
29BE EF        00056      RST      40
29BF CDDF29    05900      CALL     EXIT2         ;Get system disk
29C2 21FFFF    05910      LD      HL,-1        ;Set abort code
29C5 22C929    05920      ERREXIT LD      (RETCOD),HL
29C8 210000    05930      EXIT      LD      HL,0          ;Init to no error
29C9           05940      RETCOD EQU      $-2
29CB E5        05950      PUSH     HL
29CC FDE5      05960      PUSH     IY      ;Transfer the saved
29CE D1        05970      POP      DE          ; system DCT back

```

Format Execution Code

```

29CF 21DD2A 05980 LD HL,SYSDCT ; into the system
29D2 010A00 05990 LD BC,10 ; DCT slot
29D5 EDB0 06000 LDIR
29D7 E1 06010 POP HL
29D8 310000 06020 SPSAV LD SP,$-$ ;P/u the stack pointer
29DB 06030 @CKBRKC ;Clear break bit
29DB 3E6A 00057 LD A,106
29DD EF 00058 RST 40
29DE C9 06040 RET ; & exit to caller
06050 ;
29DF 3A2B2A 06060 EXIT2 LD A,(FMTDRV+1) ;P/u drive # just fmtd
29E2 3C 06070 INC A ;If drive never entered,
29E3 C8 06080 RET Z ; just return
29E4 3D 06090 DEC A ;If 0, we need a system
29E5 200D 06100 JR NZ,EXIT4
29E7 21EF2C 06110 LD HL,PMTSYS$ ;"load system disk..."
29EA 06120 @DSPLY
00059 IFEQ 00H,1
00060 LD HL,
00061 ENDF
29EA 3E0A 00062 LD A,10
29EC EF 00063 RST 40
29ED 06130 EXIT3 @KEY ;Request a key
29ED 3E01 00064 LD A,1
29EF EF 00065 RST 40
29F0 FE0D 06140 CP CR ;Must be <ENTER>
29F2 20F9 06150 JR NZ,EXIT3
29F4 1809 06160 EXIT4 JR RESTOR ;Restore disk to cyl 0
06170 ;
06180 ; Disk I/O requests
06190 ;
29F6 C5 06200 DRVNOP PUSH BC
29F7 AF 06210 XOR A
29F8 1830 06220 JR FMTDRV
29FA C5 06230 SELECT PUSH BC
29FB 3E01 06240 LD A,1
29FD 182B 06250 JR FMTDRV
29FF C5 06260 RESTOR PUSH BC
2A00 3E04 06270 LD A,4
2A02 1826 06280 JR FMTDRV
2A04 C5 06290 STEPIN PUSH BC
2A05 3E05 06300 LD A,5
2A07 1821 06310 JR FMTDRV
2A09 C5 06320 RSELECT PUSH BC
2A0A 3E07 06330 LD A,7
2A0C 181C 06340 JR FMTDRV
2A0E C5 06350 WRCYL PUSH BC
2A0F 3E0F 06360 LD A,15
2A11 1817 06370 JR FMTDRV
2A13 C5 06380 FMTHD PUSH BC
2A14 3E0C 06390 LD A,12
2A16 1812 06400 JR FMTDRV
2A18 C5 06410 WRSEC PUSH BC
2A19 3E0D 06420 LD A,13
2A1B 180D 06430 JR FMTDRV
2A1D C5 06440 WRSYS PUSH BC
2A1E 3E0E 06450 LD A,14
2A20 1808 06460 JR FMTDRV
2A22 C5 06470 RDSEC PUSH BC

```

Format Execution Code

```

2A23 3E09      06480      LD      A,9
2A25 1803      06490      JR      FMTDRV
2A27 C5        06500 VERSEC  PUSH   BC
2A28 3E0A      06510      LD      A,10
2A2A 0EFF      06520 FMTDRV  LD      C,-1      ;P/u drive #
2A2C C628      06530      ADD     A,40      ;Adjust SVC #
2A2E EF        06540      RST     40
2A2F C1        06550      POP     BC
2A30 C9        06560      RET
                06570 ;
                06580 ;      Perform a verification to ensure system sector
                06590 ;
2A31 CD272A    06600 VERSYS  CALL   VERSEC      ;Sector verify
2A34 2806      06610      JR      Z,VERS1    ;Bypass if not system
2A36 D606      06620      SUB     6          ;Test read system retcod
2A38 C8        06630      RET     Z          ;Go if that's what it was
2A39 C606      06640      ADD     A,6        ;Restore orig retcod
2A3B C9        06650      RET
2A3C F601      06660 VERS1   OR      1          ;S/b system, found data
2A3E 3E00      06670      LD      A,0
2A40 C9        06680      RET
                06690 ;
2A41 CD1D2A    06700 WRDIR   CALL   WRSYS      ;Write the DIR sector
2A44 C4312A    06710 WRDIR1  CALL   NZ,VERSYS  ;Verify after write
2A47 C0        06720      RET     NZ
2A48 D5        06730      PUSH   DE
2A49 0E2E      06740      LD      C,'.'      ;Display a period
2A4B          06750      @@DSP      ; for every sector written
2A4B 3E02      00066      LD      A,2
2A4D EF        00067      RST     40
2A4E D1        06760      POP     DE
2A4F C9        06770      RET
                06780 ;
                06790 ;      Routine to convert reg A to 2 decimal digits
                06800 ;
2A50 0E30      06810 CVDEC   LD      C,30H      ;Init msd to 0
2A52 D60A      06820 CVD1   SUB     10        ;Sub 10 until underflow
2A54 3803      06830      JR      C,CVD2
2A56 0C        06840      INC     C          ;Inc the count
2A57 18F9      06850      JR      CVD1
2A59 C63A      06860 CVD2   ADD     A,3AH      ;Add back 10 + '0'
2A5B 47        06870      LD      B,A        ;Lsd to B
2A5C C9        06880      RET
                06890 ;
                06900 ;      Routines to convert input strings to UC
                06910 ;      HL => Prompt string
                06920 ;
2A5D          06930 GET3    @@DSPLY      ;Display the prompt
                00068      IFEQ     00H,1
                00069      LD      HL,
                00070      ENDIF
2A5D 3E0A      00071      LD      A,10
2A5F EF        00072      RST     40
2A60 010003    06940      LD      BC,3<8      ;Init 3 keys max
2A63 1803      06950      JR      $+5
2A65 010008    06960 GET8   LD      BC,8<8      ;8-chars max
2A68 210030    06970      LD      HL,HITBUF    ;Buffer area
2A6B          06980 GET8A   @@KEYIN      ;Enter them
2A6B 3E09      00073      LD      A,9

```

Format Execution Code

```

2A6D EF      00074      RST      40
2A6E DAB929  06990      JP       C,FMTABT      ;Quit on Break
2A71 78      07000      LD       A,B          ;Get length of response
2A72 B7      07010      OR       A
2A73 C8      07020      RET       Z          ;Back if Enter only
          07030 ;
          07040 ;      Routine to convert n-character string to UC
          07050 ;
2A74 F5      07060      PUSH     AF          ;Save the registers
2A75 C5      07070      PUSH     BC
2A76 E5      07080      PUSH     HL
2A77 7E      07090 GETUC LD       A,(HL)      ;P/u a char
2A78 FE61    07100      CP       'a'          ;Skip if below 'a'
2A7A 3806    07110      JR       C,GETUC1
2A7C FE7B    07120      CP       'z'+1        ; or above 'z'
2A7E 3002    07130      JR       NC,GETUC1
2A80 CBAE    07140      RES      5,(HL)        ; else convert to UC
2A82 23      07150 GETUC1 INC      HL          ;Bump the buffer ptr
2A83 10F2    07160      DJNZ     GETUC        ;Loop thru all chars
2A85 E1      07170      POP      HL
2A86 C1      07180      POP      BC
2A87 F1      07190      POP      AF
2A88 C9      07200      RET
          07210 ;
          07220 ;      Routine to display the cylinder number
          07230 ;
2A89 C5      07240 DSPCYL PUSH     BC          ;Save ASCII cylinder #
2A8A 0E08    07250      LD       C,8          ;Back up twice &
2A8C          07260      @@DSP          ; output new position
2A8C 3E02    00075      LD       A,2
2A8E EF      00076      RST      40
2A8F 0E08    07270      LD       C,8
2A91          07280      @@DSP
2A91 3E02    00077      LD       A,2
2A93 EF      00078      RST      40
2A94 C1      07290      POP      BC          ;Recover cyl #
2A95          07300      @@DSP          ;Send MSD
2A95 3E02    00079      LD       A,2
2A97 EF      00080      RST      40
2A98 48      07310      LD       C,B
2A99          07320      @@DSP          ;Send LSD
2A99 3E02    00081      LD       A,2
2A9B EF      00082      RST      40
2A9C C9      07330      RET
          07340 ;
          07350 ;      Formatting data and tables
          07360 ;
2A9D 5E      07370 BOOTDIR DB      5EH,0,0,0,0,'BOOT'  SYS',0F6H,37H
          00 00 00 00 42 4F 4F 54
          20 20 20 20 53 59 53 F6
          37
2AAF F5      07380      DB      0F5H,9CH,5,0,0,0,0FFH,0FFH,-1,-1,-1,-1,-1,-1
          9C 05 00 00 FF FF FF
          FF FF FF FF
2ABD 5D      07390 DDIRDIR DB      5DH,0,0,0,0,'DIR'  SYS',0F6H,37H
          00 00 00 00 44 49 52 20
          20 20 20 20 53 59 53 F6
          37
2ACF 96      07400      DB      96H,42H,10,0,11H,1,0FFH,0FFH,0,0,0,0,0,0

```


Format Execution Code

```

    42 0A 00 11 01 FF FF 00
    00 00 00 00 00
0000A      07410 SYSDCT DS      10
2AE7 00      07420 STEP DFT DB      0 ;Boot step rate default
0001      07430 SECCYL DS      1 ;# of sectors per cyl
0001      07440 SECTRK DS      1 ;# of sectors per trk
          07450 ;
          07460 ; Single density 5" format table
          07470 ;
2AEA 0A      07480 S5TBL DB      10,7
          07
2AEC 00      07490 DB      0,5,1,6,2,7,3,8,4,9
          05 01 06 02 07 03 08 04
          09
2AF6 F6      07500 DB      -10,-10,-10,-10,-10,-10,-10,-10,14,0FFH
          F6 F6 F6 F6 F6 F6 F6 0E
          FF
2B00 F1      07510 DB      0F1H,6,0,1,0FEH
          06 00 01 FE
2B05 F3      07520 DB      0F3H,3,0,1,1,1,0F7H,1,0FFH,11,0FFH
          03 00 01 01 01 F7 01 FF
          0B FF
2B10 06      07530 DB      6,0,1,0FBH,0,0E5H,1,0F7H,1,0FFH,13,0FFH
          00 01 FB 00 E5 01 F7 01
          FF 0D FF
2B1C F2      07540 DB      0F2H,47H,0FFH,0F4H
          47 FF F4
2B20 00      07550 DB      0,1,2,3,4,5,6,7,8,9
          01 02 03 04 05 06 07 08
          09
          07560 ;
          07570 ; Double density 5" format table
          07580 ;
          07590 D5TBL DB      18,10
2B2A 12      0A
          0A
2B2C 00      07600 DB      0,9,1,10,2,11,3,12,4
          09 01 0A 02 0B 03 0C 04
2B35 0D      07610 DB      13,5,14,6,15,7,16,8,17
          05 0E 06 0F 07 10 08 11
2B3E EE      07620 DC      11,-18
          EE EE EE EE EE EE EE EE
          EE EE
2B49 14      07630 DB      20,4EH
          4E
2B4B F1      07640 DB      0F1H,12,0,3,0F5H,1,0FEH
          0C 00 03 F5 01 FE
2B52 F3      07650 DB      0F3H,3,0,1,1,1,0F7H,22,4EH,12,0,3,0F5H
          03 00 01 01 01 F7 16 4E
          0C 00 03 F5
2B5F 01      07660 DB      1,0FBH,0F5H,128,6DH,0B6H
          FB F5 80 6D B6
2B65 01      07670 DB      1,0F7H,1,0FFH,17,04EH
          F7 01 FF 11 4E
2B6B F2      07680 DB      0F2H,182,4EH,0F4H
          B6 4E F4
2B6F 00      07690 DB      0,1,2,3,4,5,6,7,8,9
          01 02 03 04 05 06 07 08
          09
2B79 0A      07700 DB      10,11,12,13,14,15,16,17

```

Format Execution Code

```

    0B 0C 0D 0E 0F 10 11
        07710 ;
        07720 ;      Single density 8" format table
        07730 ;
2B81 10      07740 S8TBL DB      16,2
    02
2B83 0A      07750 DB      10,5,0,11,6,1,12,7,2,13,8,3,14,9,4,15
    05 00 0B 06 01 0C 07 02
    0D 08 03 0E 09 04 0F
2B93 F0      07760 DB      -16,-16,-16,28H,0FFH
    F0 F0 28 FF
2B98 F1      07770 DB      0F1H,6,0,1,0FEH
    06 00 01 FE
2B9D F3      07780 DB      0F3H,3,0,1,1,1,0F7H,11,0FFH,6,0,1,0FBH
    03 00 01 01 01 F7 0B FF
    06 00 01 FB
2BAA 00      07790 DB      0,0E5H,1,0F7H,1,0FFH,20,0FFH
    E5 01 F7 01 FF 14 FF
2BB2 F2      07800 DB      0F2H,208,0FFH,0F4H
    D0 FF F4
2BB6 0A      07810 DB      10,0,6,12,2,8,14,4,5,11,1,7,13,3,9,15
    00 06 0C 02 08 0E 04 05
    0B 01 07 0D 03 09 0F
        07820 ;
        07830 ;      Double density 8" format table
        07840 ;
2BC6 1E      07850 D8TBL DB      30,12
    0C
2BC8 00      07860 DB      0,10,20,1,11,21,2,12,22,3,13,23,4,14,24
    0A 14 01 0B 15 02 0C 16
    03 0D 17 04 0E 18
2BD7 05      07870 DB      5,15,25,6,16,26,7,17,27,8,18,28,9,19,29
    0F 19 06 10 1A 07 11 1B
    08 12 1C 09 13 1D
2BE6 E2      07880 DC      13,-30
    E2 E2 E2 E2 E2 E2 E2 E2
    E2 E2 E2 E2
2BF3 14      07890 DB      20,4EH
    4E
2BF5 F1      07900 DB      0F1H,0CH,0,3,0F5H,1,0FEH
    0C 00 03 F5 01 FE
2BFC F3      07910 DB      0F3H,3,0,1,1,1,0F7H,22,4EH,12,0,3,0F5H
    03 00 01 01 01 F7 16 4E
    0C 00 03 F5
2C09 01      07920 DB      1,0FBH,0F5H,128,6DH,0B6H
    FB F5 80 6D B6
2C0F 01      07930 DB      1,0F7H,1,0FFH,17,4EH
    F7 01 FF 11 4E
2C15 F2      07940 DB      0F2H,0,4EH,61,4EH,0F4H
    00 4E 3D 4E F4
2C1B 00      07950 DB      0,20,11,2,22,13,4,24,15,6,26,17,8,28,19
    14 0B 02 16 0D 04 18 0F
    06 1A 11 08 1C 13
2C2A 0A      07960 DB      10,1,21,12,3,23,14,5,25,16,7,27,18,9,29
    01 15 0C 03 17 0E 05 19
    10 07 1B 12 09 1D
        07970 ;
2C39 1D      07980 FMTCYL$ DB      29,'Formatting cylinder ',3
    46 6F 72 6D 61 74 74 69

```

Format Execution Code

```

        6E 67 20 63 79 6C 69 6E
        64 65 72 20 20 20 03
2C51 1D      079900 VERCYL$ DB      29,'Verifying cylinder ',3
        56 65 72 69 66 79 69 6E
        67 20 20 63 79 6C 69 6E
        64 65 72 20 20 20 03
2C69 2A      080000 STAR$  DB      '* ',3
        20 20 20 03
2C6E 46      080100 FMTG$  DB      'Formatting... ',CR
        6F 72 6D 61 74 74 69 6E
        67 2E 2E 2E 0D
2C7C 4E      080200 NOCYL$ DB      'No cylinders available for directory',CR
        6F 20 63 79 6C 69 6E 64
        65 72 73 20 61 76 61 69
        6C 61 62 6C 65 20 66 6F
        72 20 64 69 72 65 63 74
        6F 72 79 0D
2CA1 44      080300 DIRCYL$ DB      'Directory will be placed on cylinder '
        69 72 65 63 74 6F 72 79
        20 77 69 6C 6C 20 62 65
        20 70 6C 61 63 65 64 20
        6F 6E 20 63 79 6C 69 6E
        64 65 72 20
2CC6 30      080400 DIRASC$ DB      '00',CR
        30 0D
2CC9 0A      080500 IPLSYS$ DB      LF,'Initializing DIRECTORY information: ',3
        49 6E 69 74 69 61 6C 69
        7A 69 6E 67 20 44 49 52
        45 43 54 4F 52 59 20 69
        6E 66 6F 72 6D 61 74 69
        6F 6E 3A 20 03
2CEF 0A      080600 PMTSYS$ DB      LF,'Load SYSTEM diskette <ENTER>',CR
        4C 6F 61 64 20 53 59 53
        54 45 4D 20 64 69 73 6B
        65 74 74 65 20 20 3C 45
        4E 54 45 52 3E 0D
2D0E 0A      080700 FMTCAS$ DB      LF,LF,'Formatting complete',CR
        0A 46 6F 72 6D 61 74 74
        69 6E 67 20 63 6F 6D 70
        6C 65 74 65 0D
2D24 0A      080800 FMATAB$ DB      LF,'Command aborted',CR
        43 6F 6D 6D 61 6E 64 20
        61 62 6F 72 74 65 64 0D
2D35 0A      080900 NOTFMT$ DB      LF,'Can''t, Diskette not formatted',CR
        43 61 6E 27 74 2C 20 44
        69 73 6B 65 74 74 65 20
        6E 6F 74 20 66 6F 72 6D
        61 74 74 65 64 0D
        081000 ;
        081100 ;      Patch area
        081200 ;
        2E00      081300      ORG      $<-8+1<+8
        00CB      081400 GATBUF  DS      203      ;GAT sector buffer
        2ECB 62      081500      DB      RLS,0,0,0,0 ;Ver, cyl exc, type, pswd
        00 00 00 00
        2ED0 20      081600      DB      '      MM/DD/YY'
        20 20 20 20 20 20 20 4D
        4D 2F 44 44 2F 59 59
        2EE0 00      081700      DC      32,0

```

Format Execution Code

[illegible]

Format Execution Code

4400	08440	SAFESP	EQU	\$
3000	08450		ORG	CORE\$+256
3000	08460		LORG	CORE\$+256
0100	08470	HITBUF	DS	256
	08480	;		
3100	08490		SUBTTL	'<Format Init Code>'

Format Init Code

```

3100      08510 *GET      FORMAT2:3
          03950 ;FORMAT2/ASM - Format Initialization Code
          03960 ;
          03970 ;      FORMAT routine entry point
          03980 ;
          03990 FORMAT
3100      04000      @@CKBRKC      ;Check for break
3100 3E6A      04083      LD      A,106
3102 EF      04084      RST      40
3103 2804      04010      JR      Z,FORMATA      ;Continue if no break
3105 21FFFF      04020      LD      HL,-1      ; else abort
3108 C9      04030      RET
          04040 ;
3109 ED73D929 04050 FORMATA LD      (SPSAV+1),SP      ;Save the stack pointer
310D E5      04060      PUSH     HL      ;Save cmdline ptr
310E      04070      @@DSPLY HELLO$      ;Hello message
          04085      IFEQ      01H,1
310E 21C236      04086      LD      HL,HELLO$
          04087      ENDIF
3111 3E0A      04088      LD      A,10
3113 EF      04089      RST      40
3114 CD5436      04080      CALL    GETSYS2      ;Load SYS2 overlay
          04090 ;
          04100 ;      Read config sector & extract DCT # cyls
          04110 ;
          04120      IF      @MOD4
3117 110200      04130      LD      DE,2      ;Track 0, sector 2
311A 4A      04140      LD      C,D      ;Drive 0
          04150      ENDIF
          04160 ;
          04170      IF      @MOD2
311B      04180      LD      C,0      ;Drive 0
          04190      @@GTDCT      ;Fetch DCT
          04090      LD      A,81
          04091      RST      40
          04200      LD      A,(IY+3)      ;Get dct data
          04210      AND      28H      ;Bit 5/3
          04220      CP      20H      ;8" floppy?
          04230      JR      NZ,SETSYS1      ;Go if not
          04240      LD      A,(IY+4)      ;Get data
          04250      AND      50H      ;Bit 6/4
          04260      CP      40H      ;DD not alien?
          04270      JR      NZ,SETSYS1      ;Go if not
          04280      LD      HL,HITBUF      ;Init buffer
          04290      LD      D,(IY+9)      ;Get dir cyl
          04300      LD      E,0      ;Init GAT table
311B      04310      @@RDSEC      ;Read GAT table
          04092      LD      A,49
          04093      RST      40
          04320      CP      6      ;Directory read?
          04330      JP      NZ,IOERR      ;Go on disk error
          04340      LD      A,(HITBUF+0CDH) ;Get data byte
          04350      BIT      7,A      ;System disk?
          04360      SETSYS1 LD      DE,0<8+2      ;Init cyl 0
          04370      JR      NZ,$+3      ;Go if not system
          04380      INC      D      ;Else on cyl 1
          04390      LD      C,0      ;Drive 0
          04400      ENDIF
          04410 ;
311B 210030      04420      LD      HL,HITBUF      ;Set disk buffer

```

Format Init Code

```

311E      04430      @RDSEC      ;Read sysinfo sector
311E 3E31      00094      LD      A,49
3120 EF      00095      RST      40
3121 C2A529    04440      JP      NZ,IOERR      ;Quit on read error
3124 2E76      04450      LD      L,70H+6      ;Pt to default DCTs
           04460      ;
           04470      ;      Establish the default BOOT step rate
           04480      ;
3126 E5      04490      PUSH     HL      ;Pt IY to the
3127 FDE1      04500      POP      IY      ; start of the DCTs
3129 FD7EFD    04510      LD      A,(IY+3-6)    ;P/u DCT$ default step
312C E603      04520      AND      3      ; & strip off
312E 329731    04530      LD      (STEPARM+1),A ;Keep for Step parm
           04540      ;
           04550      ;      Keep cyl count on all 8 drives
           04560      ;
3131 0608      04570      LD      B,8
3133 DD212136  04580      LD      IX,DCTCYL      ;Pt to where to stuff
3137 110A00    04590      LD      DE,10      ; 10-byte increments
313A 7E      04600 DCTL P1 LD      A,(HL)      ;P/u default # CYL
313B DD7700    04610      LD      (IX),A      ;Save in table
313E DD23      04620      INC      IX
3140 19      04630      ADD      HL,DE
3141 10F7      04640      DJNZ     DCTL P1      ;Loop for 8 DCTs
           04650      ;
3143 E1      04660      POP      HL      ;Rcvr ptr to cmdline
3144 7E      04670 FMT1 LD      A,(HL)      ;Ignore spaces
3145 23      04680      INC      HL
3146 FE20      04690      CP      ' '
3148 28FA      04700      JR      Z,FMT1
314A FE3A      04710      CP      ':'      ;Colon drive indicator?
314C 281F      04720      JR      Z,FMT2      ;Go on drive entry
           04730      ;
           04740      ;      Drive not entered, prompt for it
           04750      ;
314E 2B      04760      DEC      HL      ;Backspace command line
314F 2B      04770      DEC      HL      ; & adjust for next INC
3150 E5      04780      PUSH     HL      ;Save pointer
3151      04790 WHDRV @@DSPLY WHDRV$      ;"which drive..."
           00096      IFEQ     01H,1
3151 218937    00097      LD      HL,WHDRV$
           00098      ENDIF
3154 3E0A      00099      LD      A,10
3156 EF      00100      RST      40
3157 210030    04800      LD      HL,HITBUF      ;Input buffer for now
315A 010001    04810      LD      BC,1<8      ;Max 1 char
315D      04820      @@KEYIN      ;Get a 1-char line
315D 3E09      00101      LD      A,9
315F EF      00102      RST      40
3160 DAB929    04830      JP      C,FMTABT      ;Quit on Break
3163 7E      04840      LD      A,(HL)      ;P/u the entry
3164 D630      04850      SUB      '0'      ;Cvrt to binary
3166 FE08      04860      CP      8      ;Error if > 7
3168 30E7      04870      JR      NC,WHDRV
316A E1      04880      POP      HL      ;Rcvr command pointer
316B 1808      04890      JR      FMT2A
           04900      ;
           04910      ;      Drive entered
           04920      ;

```

Format Init Code

```

316D 7E      04930 FMT2   LD      A,(HL)      ;P/u drive #
316E D630    04940      SUB      '0'          ;Cvrt to ASCII
3170 FE08    04950      CP       8            ;Make sure not > 7
3172 D24F36  04960      JP       NC,PRMERR
3175 322B2A  04970 FMT2A LD      (FMTDRV+1),A    ;Stuff drive
3178 23      04980      INC      HL           ;Bump cmdline ptr
3179 116036  04990      LD      DE,PRMTBL$     ;Parse any parameters
317C         05000      @@PARAM
317C 3E11    00103      LD      A,17
317E EF      00104      RST      40
317F C24F36  05010      JP       NZ,PRMERR      ;Jump on parm error
          05020 ;
          05030 ;      Test if any other parm was entered
          05040 ;
3182 110000  05050 SDPARM LD      DE,0        ;Single density parm
3185 7A      05060      LD      A,D
3186 B3      05070      OR       E            ;Merge all theses parms
3187 110000  05080 DDPARM LD      DE,0        ;Double density parm
318A B2      05090      OR       D
318B B3      05100      OR       E
318C 110000  05110 SIDES LD      DE,0        ;Sides parm
318F B2      05120      OR       D
3190 B3      05130      OR       E
3191 110000  05140 CPARM LD      DE,0        ;Cylinder parm
3194 B2      05150      OR       D
3195 B3      05160      OR       E
3196 1100FF  05170 STEPARM LD      DE,0FF00H   ;Init to show if entry
3199 14      05180      INC      D            ;Did user enter it?
319A B2      05190      OR       D            ;0=no user entry
319B 32AC32  05200      LD      (PRMMRG+1),A    ;Set to non-zero if any
          05210 ;
          05220 ;      If Q-parm, then set NAME & MPW if not entered
          05230 ;
319E ED5BB132 05240      LD      DE,(QPARM+1)    ;P/u Query parm
31A2 2AEB31  05250      LD      HL,(NPARM+1)    ;P/u Name parm
31A5 7C      05260      LD      A,H
31A6 B5      05270      OR       L
31A7 2004    05280      JR       NZ,$+6        ;Go if user entered name
31A9 ED53EB31 05290      LD      (NPARM+1),DE    ; else use Q-parm value
31AD 2A3B32  05300      LD      HL,(MPARM+1)    ;P/u Password parm
31B0 7C      05310      LD      A,H
31B1 B5      05320      OR       L
31B2 2004    05330      JR       NZ,$+6        ;Go if user entered password
31B4 ED533B32 05340      LD      (MPARM+1),DE    ;Set to Q-parm entry
          05350 ;
31B8 3A2B2A  05360      LD      A,(FMTDRV+1)    ;P/u drive
31BB 4F      05370      LD      C,A            ;Set in drive register
31BC 212136  05380      LD      HL,DCTCYL      ;Find default # cyls
31BF 85      05390      ADD     A,L            ;Index the DCTCYL table
31C0 6F      05400      LD      L,A            ; according to drive #
31C1 8C      05410      ADC     A,H
31C2 95      05420      SUB     L
31C3 67      05430      LD      H,A
31C4 7E      05440      LD      A,(HL)        ;P/u cylinder count
31C5 3C      05450      INC     A            ;Offset from 1
31C6 323D33  05460      LD      (PCYL2+1),A    ;Stuff default for 5"
31C9         05470      @@GTDCT
31C9 3E51    00105      LD      A,81
31CB EF      00106      RST      40

```


Format Init Code

```

31CC FDE5      05480      PUSH    IY
31CE E1        05490      POP     HL                ;Xfer DCT to HL
31CF 11DD2A    05500      LD      DE,SYSDCT        ;Save the system's DCT
31D2 010A00    05510      LD      BC,10           ; for the drive since
31D5 EDB0      05520      LDIR                     ; we are altering it
31D7 3A1C26    05530      LD      A,(SYSPRM+1)      ;Check if "SYSTEM" parm
31DA 3C        05540      INC     A                ; entered
31DB 2007      05550      JR      NZ,FMT2B          ;Go if not
31DD FDCB035E  05560      BIT     3,(IY+3)        ;Check if hard drive
31E1 CA4936    05570      JP      Z,NOTHARD        ;Can't "SYSTEM" floppy
31E4 CDF629    05580 FMT2B CALL    DRVNOP          ;Test if drive enabled
31E7 C2A529    05590      JP      NZ,IOERR
31EA 210000    05600 NPARAM LD      HL,0                ;NAME parm entered?
31ED 7C        05610      LD      A,H
31EE B5        05620      OR     L
31EF 3C        05630      INC     A                ;Was it just NAME?
31F0 2826      05640      JR      Z,DSKNAM        ;Prompt if so
31F2 3D        05650      DEC     A                ;If entered, use it
31F3 2003      05660      JR      NZ,$+5
31F5 21063A    05670 DFTNAM LD      HL,PAKNAM$
31F8 11D02E    05680      LD      DE,GATBUF+0D0H    ;Yes, move name to field
31FB 0608      05690      LD      B,8              ;8-chars max
31FD 7E        05700 MOVNAM LD      A,(HL)          ;P/u a char
31FE FE22      05710      CP      ""              ;Closing "
3200 2829      05720      JR      Z,CKNAME        ;Exit if end of parm
3202 FE20      05730      CP      20H            ;Permit all but controls
3204 DA2B32    05740      JP      C,CKNAME
3207 FE61      05750      CP      'a'              ;If char is lower case,
3209 3806      05760      JR      C,MOVNAM1
320B FE7B      05770      CP      'z'+1
320D 3002      05780      JR      NC,MOVNAM1
320F EE20      05790      XOR     20H              ; make it UC
3211 12        05800 MOVNAM1 LD      (DE),A          ;Put char in buffer
3212 23        05810      INC     HL              ;Bump both ptrs
3213 13        05820      INC     DE
3214 10E7      05830      DJNZ   MOVNAM          ;Loop til complete
3216 1813      05840      JR      CKNAME          ;Check if valid name
3216 1813      05850      ;
3216 1813      05860      ; Prompt user for name parameter
3216 1813      05870      ;
3218          05880 DSKNAM @@DSPLY DSKNAM$          ;"diskette name?"
3218          00107 IFEQ    01H,1
3218 21A637    00108 LD      HL,DSKNAM$
3218          00109 ENDIF
321B 3E0A      00110 LD      A,10
321D EF        00111 RST     40
321E CD652A    05890      CALL    GET8              ;Get 8 chars, make UC
3221 28D2      05900      JR      Z,DFTNAM        ;Use default if no entry
3223 48        05910      LD      C,B              ;Only move to name field
3224 0600      05920      LD      B,0              ; how many were entered
3226 11D02E    05930      LD      DE,GATBUF+0D0H
3229 EDB0      05940      LDIR
322B 11D02E    05950 CKNAME LD      DE,GATBUF+0D0H    ;Now check if illegal
322E CDBC35    05960      CALL    CKMPW0           ; chars in name
3231 C24136    05970      JP      NZ,BADNAM        ; & quit if so
3234 21D82E    05980 GETDAT LD      HL,GATBUF+0D8H    ;Get today's date & stuff
3237          05990      @@DATE
3237 3E12      00112 LD      A,18
3239 EF        00113 RST     40

```

Format Init Code

```

06000 ;
06010 ; Master Password handling
06020 ;
323A 210000 06030 MPARM LD HL,0 ;Did user enter the MPW?
323D 7C 06040 LD A,H
323E B5 06050 OR L
323F 3C 06060 INC A ;If only MPW, then prompt
3240 2821 06070 JR Z,MPW ;Go prompt if not
3242 3D 06080 DEC A
3243 2003 06090 JR NZ,$+5 ;If entered, use it
3245 210E3A 06100 DFTMPW LD HL,PAKMPW$ ; else use ours
3248 115736 06110 LD DE,MPWBUF ;Shift to pswd field
324B 0608 06120 LD B,8
324D 7E 06130 MOVMPW LD A,(HL)
324E FE30 06140 CP 30H ;No spaces permitted
3250 3819 06150 JR C,PRSMWP ;End also on closing "
3252 FE61 06160 CP 'a' ;Need cvrt to UC?
3254 3806 06170 JR C,MOVMPW1
3256 FE7B 06180 CP 'z'+1
3258 3002 06190 JR NC,MOVMPW1
325A EE20 06200 XOR 20H ;Cvrt to UC
325C 12 06210 MOVMPW1 LD (DE),A ;Store the char and
325D 13 06220 INC DE ; bump the buffer ptrs
325E 23 06230 INC HL
325F 10EC 06240 DJNZ MOVMPW
3261 1808 06250 JR PRSMWP ;Check if valid password
06260 ;
06270 ; Prompt for master password
06280 ;
3263 21B737 06290 MPW LD HL,MPW$ ;"master..."
3266 CD9535 06300 CALL INPMPW
3269 30DA 06310 JR NC,DFTMPW ;Use default on <ENTER>
06320 ;
06330 ; Parse the password & stuff into GAT sector buffer
06340 ;
326B 115736 06350 PRSMWP LD DE,MPWBUF
326E CDB535 06360 CALL CKMPW ;Check for valid MPW
3271 C2A529 06370 JP NZ,IOERR
3274 22CE2E 06380 LD (GATBUF+0CEH),HL ;Stuff it
3277 FDCB0466 06390 BIT 4,(IY+4) ;Jump if alien controller
327B C2E133 06400 JP NZ,CALCGPC
327E 212936 06410 LD HL,TBLDATA ;Pt to config tables
3281 110600 06420 LD DE,6 ;Index the table
3284 FDCB036E 06430 BIT 5,(IY+3) ;8" drive?
3288 2802 06440 JR Z,INITDEN ;Bypass if not
328A 19 06450 ADD HL,DE ; else move to 8" configs
328B 19 06460 ADD HL,DE
328C 22EF32 06470 INITDEN LD (SETSDEN+1),HL ; & stuff for SDEN option
328F EB 06480 EX DE,HL ;6->HL, SDEN->DE
3290 19 06490 ADD HL,DE ;Pt to DDEN index table
3291 22DF32 06500 LD (SETDDEN+1),HL ;Stuff DDEN config ptr
3294 EB 06510 EX DE,HL ;HL=SDEN, DE=DDEN
3295 FDCB03B6 06520 RES 6,(IY+3) ;Set DCT to SDEN
3299 FDCB0476 06530 BIT 6,(IY+4) ;Test if DDEN capability
329D 2805 06540 JR Z,SETSTD ;Go if single
329F EB 06550 EX DE,HL ;HL->DDEN table
32A0 FDCB03F6 06560 SET 6,(IY+3) ;Set DCT to DDEN
32A4 CD5935 06570 SETSTD CALL SETUP ;Init to std config
32A7 FDCB04AE 06580 RES 5,(IY+4) ;Set to 1-sided

```

Format Init Code

```

32AB 3E00      06590 PRMMRG LD      A,0          ;<>0 if config parms
32AD B7        06600      OR      A              ; in command line
32AE 2008      06610      JR      NZ,GETDEN
32B0 11FFFF    06620 QPARM LD      DE,-1        ;Prompts? Default=Y
32B3 7A        06630      LD      A,D
32B4 B3        06640      OR      E
32B5 CAD633    06650      JP      Z,PSTEP1        ;Go if no prompting
32B8 FDCB0476  06660 GETDEN BIT     6,(IY+4)      ;Bypass DDEN request msg
32BC 283A      06670      JR      Z,PMTSIDE       ; if no DDEN capability
32BE 3AAC32    06680      LD      A,(PRMMRG+1)     ;Also, don't prompt if
32C1 B7        06690      OR      A              ; any config parm was
32C2 2013      06700      JR      NZ,GDDEN1       ; entered with command
32C4 216538    06710      LD      HL,DEN?$        ;Density <S,D>...
32C7 CD5D2A    06720      CALL    GET3
32CA 282C      06730      JR      Z,PMTSIDE       ;Go on <ENTER>
32CC 7E        06740      LD      A,(HL)          ;P/u response
32CD FE53      06750      CP      'S'            ;Single Density?
32CF 281D      06760      JR      Z,SETSDEN
32D1 FE44      06770      CP      'D'            ;Double density?
32D3 2809      06780      JR      Z,SETDDEN
32D5 18E1      06790      JR      GETDEN          ;Redo if bad response
32D7 3A8831    06800 GDDEN1 LD      A,(DDPARM+1)   ;Not prompted, was DDEN
32DA EEFF      06810      XOR      -1            ; set in command line?
32DC 2009      06820      JR      NZ,GSDEN1       ;Bypass if not
32DE 210000    06830 SETDDEN LD      HL,$-$       ;P/u DDEN index table
32E1 FDCB03F6  06840      SET      6,(IY+3)       ;Set DCT to DDEN
32E5 180E      06850      JR      CHGDEN
32E7 3A8331    06860 GSDEN1 LD      A,(SDPARM+1)   ;Was SDEN parm
32EA EEFF      06870      XOR      -1            ; on command line?
32EC 200A      06880      JR      NZ,PMTSIDE       ;Go if not
32EE 210000    06890 SETSDEN LD      HL,$-$       ;P/u SDEN index table
32F1 FDCB03B6  06900      RES      6,(IY+3)       ;Set DCT to SDEN
32F5 CD5935    06910 CHGDEN CALL    SETUP        ;Init #CYLs & alloc
32F8 3AAC32    06920 PMTSIDE LD      A,(PRMMRG+1)  ;Config parms entered
32FB B7        06930      OR      A              ;On command line?
32FC 2020      06940      JR      NZ,PMTS1        ;Bypass if yes
32FE FDE5      06950      PUSH    IY            ;P/u flag table
3300           06960      @@FLAGS                ; and check if
3300 3E65      00114      LD      A,101
3302 EF        00115      RST      40
3303 FDCB0B6E  06970      BIT      5,(IY+'L'-'A')   ; 2-side inhibit?
3307 FDE1      06980      POP      IY
3309 2013      06990      JR      NZ,PMTS1        ;If set, use 1 side
330B 214638    07000      LD      HL,SIDES$       ;"double sided...?
330E CD5D2A    07010      CALL    GET3          ;Get # sides wanted
3311 2816      07020      JR      Z,PMTCYL        ;Go on <ENTER>
3313 7E        07030      LD      A,(HL)          ;P/u response char
3314 FE31      07040      CP      '1'            ;1 is ok
3316 2811      07050      JR      Z,PMTCYL
3318 FE32      07060      CP      '2'            ; and so is 2
331A 20DC      07070      JR      NZ,PMTSIDE       ; but redo on anything else
331C 1805      07080      JR      TSTSID
331C           07090 ;
331C           07100 ;      Check side parm from command line
331C           07110 ;
331E 3A8D31    07120 PMTS1 LD      A,(SIDES+1)     ;How many sides?
3321 FE02      07130      CP      2
3323 2004      07140 TSTSID JR      NZ,PMTCYL     ;DCT ok if not 2
3325 FDCB04EE  07150      SET      5,(IY+4)       ;Set 2-sided drive

```

Format Init Code

```

3329 FD7E03 07160 PMTCYL LD A,(IY+3) ;No cylinder request
332C E628 07170 AND 28H ; if either hard drive
332E 2033 07180 JR NZ,PMTSTEP ; or 8" drive
3330 3AAC32 07190 PCYL1 LD A,(PRMMRG+1) ;P/u config test byte &
3333 B7 07200 OR A ; bypass cyl req if user
3334 2019 07210 JR NZ,PCYL4 ; entered cmd line parms
3336 21CA37 07220 LD HL,NUMCYL$ ;"number of cyls..?
3339 CD5D2A 07230 CALL GET3
333C 3E00 07240 PCYL2 LD A,0 ;P/u default # cyls
333E C48235 07250 CALL NZ,CVBIN ;Get # of cyls on CR
3341 FE61 07260 PCYL3 CP 96+1 ;System cannot support
3343 30EB 07270 JR NC,PCYL1 ; anything over 96 (95)
3345 FE23 07280 CP 35
3347 38E7 07290 JR C,PCYL1 ;Must be 35 or more
3349 3D 07300 DEC A ;Adjust to zero offset
334A FD7706 07310 LD (IY+6),A ; & stuff in DCT
334D 1814 07320 JR PMTSTEP
07330 ;
07340 ; User entered config parms with command line
07350 ;
334F 3A9231 07360 PCYL4 LD A,(CPARM+1) ;Was cyl= one of them?
3352 B7 07370 OR A
3353 280E 07380 JR Z,PMTSTEP ;Bypass if not
3355 FE61 07390 CP 96+1
3357 D24F36 07400 JP NC,PRMERR ;Parm error if too big
335A FE23 07410 CP 35
335C DA4F36 07420 JP C,PRMERR ; or too small
335F 3D 07430 DEC A ;Adjust to zero offset
3360 FD7706 07440 LD (IY+6),A ; & stuff into DCT
3363 FDCB0466 07450 PMTSTEP BIT 4,(IY+4) ;Alien controller?
3367 208F 07460 JR NZ,PMTSIDE ;No adjustable step rate if so
07470 ;
07480 ; If step rate parm wasn't entered, prompt
07490 ; for it but first determine 8" or 5" drive
07500 ;
3369 3AAC32 07510 LD A,(PRMMRG+1) ;Did user enter config
336C B7 07520 OR A ;Parms on command line?
336D 2067 07530 JR NZ,PSTEP1 ;Go to step prompt if yes
07540 ;
336F FDE5 07550 PUSH IY ;P/u flag table and
3371 07560 @@FLAGS ; check if
3371 3E65 00116 LD A,101
3373 EF 00117 RST 40
3374 FDCB0B46 07570 BIT 0,(IY+'L'-'A') ; step prompt inhibited
3378 FDE1 07580 POP IY
337A 205A 07590 JR NZ,PSTEP1 ;Bypass if set
07600 ;
337C FDCB036E 07610 BIT 5,(IY+3) ;Need prompt, 8"?
3380 202A 07620 JR NZ,STEP8 ;Jump if 8"
07630 ;
07640 ; 5" drive step rate parsing
07650 ;
3382 21E137 07660 STEP5 LD HL,STEP5$ ;"...step rate - 5"
3385 CD5D2A 07670 CALL GET3
3388 CD8235 07680 CALL CVBIN ;Get 5" step rate
338B B7 07690 OR A ;Use default?
338C 2848 07700 JR Z,PSTEP1 ;Go if parm not entered
338E 0600 07710 LD B,0 ;Init key to 0
3390 FE06 07720 CP 6

```

Format Init Code

```

3392 2849      07730      JR      Z,GOTSTEP
3394 0601      07740      LD      B,1          ;Init key to 1
3396 FE0C      07750      CP      12
3398 2843      07760      JR      Z,GOTSTEP
339A 0602      07770      LD      B,2          ;Init key to 2
339C FE14      07780      CP      20
339E 283D      07790      JR      Z,GOTSTEP
33A0 0603      07800      LD      B,3          ;Init key to 3
33A2 FE1E      07810      CP      30
33A4 2837      07820      JR      Z,GOTSTEP
33A6 FE28      07830      CP      40
33A8 2833      07840      JR      Z,GOTSTEP
33AA 18D6      07850      JR      STEP5          ;Re-request, bad value
              07860      ;
              07870      ;      8" drive step rate parsing
              07880      ;
33AC 211338    07890 STEP8 LD      HL,STEP8$      ;"step rate - 8"...
33AF CD5D2A    07900      CALL     GET3
33B2 CD8235    07910      CALL     CVBIN          ;Get 8" step rate
33B5 B7        07920      OR      A              ;Use default?
33B6 281E      07930      JR      Z,PSTEP1      ;Go if not entered
33B8 0600      07940      LD      B,0          ;Init key to 0
33BA FE03      07950      CP      3
33BC 281F      07960      JR      Z,GOTSTEP
33BE 0601      07970      LD      B,1          ;Init key to 1
33C0 FE06      07980      CP      6
33C2 2819      07990      JR      Z,GOTSTEP
33C4 0602      08000      LD      B,2          ;Init key to 2
33C6 FE0A      08010      CP      10
33C8 2813      08020      JR      Z,GOTSTEP
33CA 0603      08030      LD      B,3          ;Init key to 3
33CC FE0F      08040      CP      15
33CE 280D      08050      JR      Z,GOTSTEP
33D0 FE14      08060      CP      20
33D2 2809      08070      JR      Z,GOTSTEP
33D4 18D6      08080      JR      STEP8          ;Bad entry, re-request
33D6 3A9731    08090 PSTEP1 LD      A,(STEPARM+1) ;P/u step parm entry
33D9 E603      08100      AND     3              ;Keep 2 lo-order bits
33DB 1801      08110      JR      $+3
33DD 78        08120 GOTSTEP LD      A,B          ;Stuff boot step rate key
33DE 32E72A    08130      LD      (STEPDFT),A
              08140      ;
              08150      ;      Routine to calculate the # of grans per logical
              08160      ;      cylinder so that the GAT byte can be constructed
              08170      ;
33E1 FD7E08    08180 CALCGPC LD      A,(IY+8)      ;P/u # of grans per cyl
33E4 07        08190      RLCA          ;Rotate to bits 0-2
33E5 07        08200      RLCA
33E6 07        08210      RLCA
33E7 E607      08220      AND     7              ;Strip off other data
33E9 3C        08230      INC     A              ;Adj for zero offset
              08240      ;
              08250      ;      If double siding (cylindering), double the count
              08260      ;
33EA FDCB046E  08270      BIT     5,(IY+4)      ;Test if 2-sided drive
33EE 2801      08280      JR      Z,$+3          ;Bypass if only 1-sided
33F0 87        08290      ADD     A,A          ;Double the grans/cyl
33F1 01FFFF    08300      LD      BC,0FFFFH      ;Init GAT byte to ones
33F4 CB20      08310 CGPC1 SLA     B              ;Now keep removing low

```

Format Init Code

Address	Label	Hex	Op	Opnd	Comment
33F6	3D	08320	DEC	A	; order bits , 1 bit for
33F7	20FB	08330	JR	NZ,CGPC1	; each available granule
33F9	21002E	08340	LD	HL,GATBUF	;Pt to GAT buffer area
33FC	FD7E06	08350	LD	A,(IY+6)	;P/u highest # cylinder
33FF	70	08360	LD	(HL),B	;Stuff the GAT byte into
3400	2C	08370	INC	L	;Each position of the GAT
3401	BD	08380	CP	L	;One byte per cylinder
3402	30FB	08390	JR	NC,CGPC2	
		08400			
		08410			
		08420			
		08430			
		08440			
3404	3ECB	08440	LD	A,0CBH	;Continue to stuff GAT
3406	11	08450	DB	11H	; until cyl 202
3407	71	08460	LD	(HL),C	;Use FFH to show unused
3408	2C	08470	INC	L	
3409	BD	08480	CP	L	;First test here for
340A	20FB	08490	JR	NZ,CGPC3	; match against 202
		08500			
		08510			
		08520			
		08530			
340C	3A2B2A	08530	LD	A,(FMTDRV+1)	;P/u drive
340F	B7	08540	OR	A	
3410	2020	08550	JR	NZ,PFMT1	;Bypass if other than 0
3412		08560	@@DSPLY	PMTDST\$; "load dest disk...
		00118	IFEQ	01H,1	
3412	21C038	00119	LD	HL,PMTDST\$	
		00120	ENDIF		
3415	3E0A	00121	LD	A,10	
3417	EF	00122	RST	40	
3418	FDE5	08570	PUSH	IY	;Save DCT pointer
341A		08580	@@FLAGS		;Point to flags
341A	3E65	00123	LD	A,101	
341C	EF	00124	RST	40	
341D	FDCB126E	08590	BIT	5,(IY+'S'-'A')	;Check for JCL active
3421	FDE1	08600	POP	IY	;Restore pointer
3423	C2B929	08610	JP	NZ,FMTABT	;Abort if in JCL
3426	210030	08620	LD	HL,HITBUF	
3429	010000	08630	LD	BC,0	;Zero characters means
342C		08640	@@KEYIN		;Enter or Break only
342C	3E09	00125	LD	A,9	
342E	EF	00126	RST	40	
342F	DAB929	08650	JP	C,FMTABT	;Abort if Break
3432	FDE5	08660	PUSH	IY	;Xfer DCT ptr to HL
3434	E1	08670	POP	HL	; & move DCT again
3435	111736	08680	LD	DE,TMPDCT	; to store tempy
3438	010A00	08690	LD	BC,10	
343B	EDB0	08700	LDIR		
		08710	IF	@MOD2	
		08720	CALL	SELECT	
		08730	JP	NZ,IOERR	;Go on error
		08740	ENDIF		
343D	CDFF29	08750	CALL	RESTOR	;Restore to cyl 0
3440	C2A529	08760	JP	NZ,IOERR	;Go on error
3443	CD092A	08770	CALL	RSELCT	;Reselect drive
3446	C2A529	08780	JP	NZ,IOERR	;Go on error
3449	FDCB0466	08790	BIT	4,(IY+4)	;Jump if alien controller
344D	2040	08800	JR	NZ,PFMT3	
344F	218738	08810	LD	HL,NOTRDY\$;Init "drive not ready

Format Init Code

```

3452 CB7F      08820      BIT      7,A          ;Test FDC status for READY
3454 C2BC29    08830      JP       NZ,EXTERR    ;Quit if not ready
3457 21AC38    08840      LD       HL,NODRV$     ;Init "drive not in...
345A CB57      08850      BIT      2,A          ;Test FDC status for TRACK-0
345C CABC29    08860      JP       Z,EXTERR    ; & error if not at track 0
345F CDEC35    08870      CALL     CKDRV      ;Ck if floppy not present
3462 20AE      08880      JR       NZ,PMTDST
3464 219738    08890      LD       HL,CANTWR$    ;Init "write protected..
3467 07        08900      RLCA          ;Align to bit 7
3468 FDB603    08910      OR        (IY+3)      ;Combine with soft WP
346B E680      08920      AND       80H        ;WP error?
346D C2BC29    08930      JP       NZ,EXTERR    ;Can't format over WP
3470 3A1C26    08940      LD       A,(SYSPRM+1) ;Don't check space needed
3473 B7        08950      OR        A          ; if SYSTEM info only
3474 2019      08960      JR       NZ,PFMT3
3476 210031    08970      LD       HL,FORMAT    ;Start of format buffer
3479 110000    08980      LD       DE,0      ;P/u format space needed
347C 19        08990      ADD      HL,DE      ;Pt to last addr needed
347D 54        09000      LD       D,H        ;Xfer to reg DE
347E 5D        09010      LD       E,L
347F 210000    09020      LD       HL,0      ;Set up for HIGH$ fetch
3482 45        09030      LD       B,L
3483           09040      @@HIGH$      ;Make sure it won't wrap
3483 3E64      09127      LD       A,100
3485 EF        09128      RST       40
3486 AF        09050      XOR      A
3487 ED52      09060      SBC      HL,DE      ; into protected memory
3489 216037    09070      LD       HL,NOMEM$    ;Init "insufficient mem..
348C DABC29    09080      JP       C,EXTERR    ;Quit if no memory available
348F 110000    09090      LD       DE,0      ;Init to cyl 0, sect 0
3492 CD272A    09100      CALL     VERSEC      ;Verify BOOT
3495 C24535    09110      JP       NZ,PFMT6    ;Assume unformatted if err
                 09120 ;
                 09130 ;
                 09140 ;
3498 3A1C26    09150      LD       A,(SYSPRM+1) ;Ignore data if SYSTEM
349B B7        09160      OR        A          ; info only
349C C24535    09170      JP       NZ,PFMT6
349F 210030    09180      LD       HL,HITBUF    ;Pt to i/o buffer
34A2 CD222A    09190      CALL     RDSEC      ;Now try to read BOOT
34A5 C2A529    09200      JP       NZ,IOERR    ;Jump on error
34A8           09210      @@LOGOT   HASDAT$    ;Show "disk contains data
                 09129      IFEQ      01H,1
34A8 21E338    09130      LD       HL,HASDAT$
                 09131      ENDIF
34AB 3E0C      09132      LD       A,12
34AD EF        09133      RST       40
34AE 21FA38    09220      LD       HL,NOFMT$    ;Init "non-std format
                 09230 ;
                 09240 ;
                 09250 ;
34B1 3A0230    09260      LD       A,(HITBUF+2) ;P/u dir cyl # (possible)
34B4 FDBE06    09270      CP        (IY+6)      ;Check against max cyl #
34B7 3069      09280      JR       NC,PFMT5    ;Go if bigger (or =)
                 09290 ;
                 09300 ;
                 09310 ;
34B9 210030    09320      LD       HL,HITBUF
34BC 5D        09330      LD       E,L

```

Appears formatted, is there SYSTEM information?

BOOT was read, is there a valid directory pointer

Read the assumed GAT & test it

Format Init Code

```

34BD 57      09340      LD      D,A          ;Pt to assumed GAT sector
34BE 210030  09350      LD      HL,HITBUF      ;Pt to buffer
34C1 CD222A  09360      CALL    RDSEC          ;Read the sector
34C4 FE06    09370      CP        6            ;Dir errcod returned?
34C6 2805    09380      JR        Z,PFMT4       ;Jump if yes & grab data
34C8 210E39  09390      LD      HL,CANTRD$      ;Init "unreadable dir..."
34CB 1855    09400      JR        PFMT5
34CD 212339  09410 PFMT4 LD      HL,NODIR$      ;Init "non-init dir"
34D0 3ADA30  09420      LD      A,(HITBUF+0DAH) ;Check if date field
34D3 FE2F    09430      CP        '/'          ; is present
34D5 204B    09440      JR        NZ,PFMT5       ;Jump if no
              09450 ;
              09460 ;
              09470 ;
              09480 ;
              09490 ;
              09500 ;
              09510 ;
              09520 ;
              09530 ;
              09540 ;
              09550 ;
              09560 ;
              09570 ;
              09580 ;
              09590 ;
              09600 ;
              09610 ;
              09620 ;
              09630 ;
              09640 ;
              09650 ;
              09660 ;
              09670 ;
              09680 ;
              09690 ;
              09700 ;
              09710 ;
              09720 ;
              09730 ;
              09740 ;
              09750 ;
              09760 ;
              09770 ;
              09780 ;
              09790 ;
              09800 ;
              09810 ;
              09820 ;
              09830 ;
              09840 ;
              09850 ;

34D7 21D030  09480      LD      HL,HITBUF+0D0H
34DA 114239  09490      LD      DE,PACKID$+5      ;Move name & date into
34DD 010800  09500      LD      BC,8            ; display message field
34E0 EDB0    09510      LDIR
34E2 115139  09520      LD      DE,PACKID$+14H
34E5 0E08    09530      LD      C,8
34E7 EDB0    09540      LDIR

              09550 ;
              09560 ;
              09570 ;
              09580 ;
              09590 ;
              09600 ;
              09610 ;
              09620 ;
              09630 ;
              09640 ;
              09650 ;
              09660 ;
              09670 ;
              09680 ;
              09690 ;
              09700 ;
              09710 ;
              09720 ;
              09730 ;
              09740 ;
              09750 ;
              09760 ;
              09770 ;
              09780 ;
              09790 ;
              09800 ;
              09810 ;
              09820 ;
              09830 ;
              09840 ;
              09850 ;

34E9 2ACE30  09580      LD      HL,(HITBUF+0CEH)      ;P/u disk MPW
34EC 11E042  09590      LD      DE,PASSWORD      ;Password=PASSWORD
34EF AF      09600      XOR        A
34F0 ED52    09610      SBC      HL,DE          ;Is it password?
34F2 213D39  09620      LD      HL,PACKID$      ;Init "Name=", Date=
34F5 282B    09630      JR        Z,PFMT5       ;If match, go check ABS
34F7        09640      @@LOGOT
              09650 ;
              09660 ;
              09670 ;
              09680 ;
              09690 ;
              09700 ;
              09710 ;
              09720 ;
              09730 ;
              09740 ;
              09750 ;
              09760 ;
              09770 ;
              09780 ;
              09790 ;
              09800 ;
              09810 ;
              09820 ;
              09830 ;
              09840 ;
              09850 ;

34F7 3E0C    096137     LD      A,12
34F9 EF      096138     RST      40
34FA FDE5    09650     PUSH     IY            ;Abort if in JCL
34FC        09660     @@FLAGS
34FC 3E65    096139     LD      A,101
34FE EF      096140     RST      40
34FF FDCB126E 09670     BIT      5,(IY+'S'-'A') ;Test if "DOing"
3503 FDE1    09680     POP      IY
3505 C2B929  09690     JP        NZ,FMTABT      ;Can't get PW if in JCL
              09700 ;
              09710 ;
              09720 ;
              09730 ;
              09740 ;
              09750 ;
              09760 ;
              09770 ;
              09780 ;
              09790 ;
              09800 ;
              09810 ;
              09820 ;
              09830 ;
              09840 ;
              09850 ;

3508 215A39  09730 OLDMPW LD      HL,OLDMPW$      ;"What's the old MPW?"
350B CD9535  09740      CALL    INMPW          ;Grab user input to match
350E 30F8    09750      JR        NC,OLDMPW
3510 115736  09760      LD      DE,MPWBUF
3513 CDB935  09770      CALL    HASHMPW          ;Hash user entry
              09780 ;
              09790 ;
              09800 ;
              09810 ;
              09820 ;
              09830 ;
              09840 ;
              09850 ;

3516 EB      09810      EX        DE,HL          ;Xfer hashed MPW to DE
3517 2ACE30  09820      LD      HL,(HITBUF+0CEH)      ;Else grab pack MPW
351A AF      09830      XOR        A            ;Clear carry flag
351B ED52    09840      SBC      HL,DE          ;Did user enter pack MPW?
351D C24536  09850      JP        NZ,BADMPW      ;Abort if no match

```


Format Init Code

```

3520 1823      009860      JR      PFMT6
                009870 ;
                009880 ;      The directory was not readable - req assurance
                009890 ;
3522          009900 PFMT5  @@LOGOT
                00141      IFEQ      00H,1
                00142      LD        HL,
                00143      ENDIF
3522 3E0C      00144      LD        A,12
3524 EF        00145      RST        40
3525 110000    009910 APARM LD        DE,0      ;ABS parameter
3528 1C        009920      INC        E
3529 281A      009930      JR        Z,PFMT6      ;Go if ABS used
352B FDE5      009940      PUSH       IY
352D          009950      @@FLAGS
352D 3E65      00146      LD        A,101
352F EF        00147      RST        40
3530 FDCB126E 009960      BIT        5,(IY+'S'-'A') ;Test if "DOing"
3534 FDE1      009970      POP        IY
3536 C2B929    009980      JP        NZ,FMTABT      ;Abort if JCL but no ABS
3539 21B439    009990      LD        HL,SURE?$(      ;"are you sure...?"
353C CD5D2A    100000      CALL       GET3          ;Get response
353F 7E        100010      LD        A,(HL)
3540 FE59      100020      CP        'Y'          ;If not Yes, abort
3542 C2B929    100030      JP        NZ,FMTABT
3545 FDE5      100040 PFMT6  PUSH       IY          ;Move drive code table
3547 D1        100050      POP        DE          ; back into place
3548 211736    100060      LD        HL,TMPDCT      ; into system slot
354B 010A00    100070      LD        BC,10
354E EDB0      100080      LDIR
3550 CDFF29    100090      CALL       RESTOR        ;Restore to cylinder 0
3553 C2A529    101000      JP        NZ,IOERR        ;Go on error
3556 C30126    101100      JP        GOFMT          ;Go and format it
                10120 ;
                10130 ;      Routine to set up the DCT for format
                10140 ;
3559 3A3D33    101500 SETUP LD        A,(PCYL2+1)      ;P/u the highest # cyl
355C FDCB036E 101600      BIT        5,(IY+3)        ;If 8" drive, use 77
3560 2802      101700      JR        Z,$+4          ;Go if only 5"
3562 3E4D      101800      LD        A,77          ;8" drives are 77 cyls
3564 3D        101900      DEC        A
3565 FD7706    102000      LD        (IY+6),A        ;Stuff in our DCT
3568 5E        102100      LD        E,(HL)        ;Grab address to
3569 23        102200      INC        HL          ; master formatting table
356A 56        102300      LD        D,(HL)
356B 23        102400      INC        HL
356C ED530926 102500      LD        (FMTTBL+1),DE    ;Stuff for later use
3570 5E        102600      LD        E,(HL)        ;P/u DCT+7 data
3571 23        102700      INC        HL          ;Max sector, # of heads
3572 56        102800      LD        D,(HL)        ;P/u DCT+8 data, # of
3573 23        102900      INC        HL          ; sectors/gran & grans/cyl
3574 FD7307    103000      LD        (IY+7),E        ;Stuff these values into
3577 FD7208    103100      LD        (IY+8),D        ; our DCT
357A 5E        103200      LD        E,(HL)        ;P/u space needed for
357B 23        103300      INC        HL          ; the formatting buffer
357C 56        103400      LD        D,(HL)
357D ED537A34 103500      LD        (PFMT2+1),DE    ; & stuff that for later
3581 C9        103600      RET
                10370 ;

```

Format Init Code

```

10380 ;          Convert decimal ASCII to binary
10390 ;
3582 1E00      10400 CVBIN  LD      E,0          ;Init value to 0
3584 7E        10410 CVB1  LD      A,(HL)        ;Get a character
3585 23        10420      INC      HL            ;Bump buff ptr
3586 D630      10430      SUB      30H          ;Make binary
3588 47        10440      LD      B,A
3589 FE0A      10450      CP      0AH          ;Was it a decimal digit?
358B 7B        10460      LD      A,E
358C D0        10470      RET      NC          ;Return if not
358D 87        10480      ADD      A,A          ;Mult previous value X 10
358E 87        10490      ADD      A,A
358F 83        10500      ADD      A,E
3590 87        10510      ADD      A,A
3591 80        10520      ADD      A,B          ;Add in new digit
3592 5F        10530      LD      E,A          ;Put results in E
3593 18EF      10540      JR      CVB1          ;Loop
10550 ;
3595          10560 INMPW  @@DSPLY
10560 IFEQ 00H,1
10560 LD HL,
10560 ENDF
3595 3E0A      00151      LD      A,10
3597 EF        00152      RST      40
3598 215736    10570      LD      HL,MPWBUF      ;Use this buffer
359B 0608      10580      LD      B,8          ;8 chars max
359D CD6B2A    10590      CALL   GET8A          ;Input the pswd
35A0 C8        10600      RET      Z          ;Go if Enter only
35A1 EB        10610      EX      DE,HL
35A2 83        10620      ADD      A,E          ;Find where the X'0D' was
35A3 6F        10630      LD      L,A          ; stuffed & cover it
35A4 7A        10640      LD      A,D
35A5 CE00      10650      ADC      A,0
35A7 67        10660      LD      H,A
35A8 3E08      10670      LD      A,8          ;If 8 chars entered,
35AA 90        10680      SUB      B
35AB 37        10690      SCF
35AC C8        10700      RET      Z          ; done
35AD 47        10710      LD      B,A          ; else pad the buffer
35AE 3620      10720 FILLBLK LD      (HL),' '    ; w/spaces
35B0 23        10730      INC      HL
35B1 10FB      10740      DJNZ    FILLBLK
35B3 37        10750      SCF
35B4 C9        10760      RET
10770 ;
35B5 CDBC35    10780 CKMPW  CALL   CKMPW0
35B8 C0        10790      RET      NZ
10800 ;
10810 ;          Hash a diskette password
10820 ;
35B9 3EE4      10830 HASHMPW LD      A,0E4H      ;Use SYS2 routine
35BB EF        10840      RST      40
10850 ;
35BC 0608      10860 CKMPW0 LD      B,8          ;8 char to check
35BE D5        10870      PUSH    DE          ;Xfer start of PW
35BF E1        10880      POP     HL           ; to HL
35C0 7E        10890      LD      A,(HL)       ;P/u 1st char
35C1 180E      10900      JR      CKMPW2       ; & check <A-Z>
35C3 23        10910 CKMPW1 INC      HL        ;Advance to next char

```

Format Init Code

```

35C4 7E      10920      LD      A,(HL)      ;P/u the char
35C5 FE20    10930      CP      ' '
35C7 2818    10940      JR      Z,CKMPW7      ;Go on space
35C9 FE30    10950      CP      '0'
35CB 3818    10960      JR      C,INVMPW      ;Bad if less than 0
35CD FE3A    10970      CP      '9'+1      ; or greater than 9
35CF 3808    10980      JR      C,CKMPW3
35D1 FE41    10990 CKMPW2 CP      'A'
35D3 3810    11000      JR      C,INVMPW      ; but less than A
35D5 FE5B    11010      CP      'Z'+1
35D7 300C    11020      JR      NC,INVMPW     ;More than Z also bad
35D9 10E8    11030 CKMPW3 DJNZ     CKMPW1     ;Char ok, do another
35DB AF      11040      XOR      A          ;Set Z, PW good
35DC C9      11050      RET
                11060 ;
35DD 23      11070 CKMPW5 INC      HL          ;Next char position
35DE BE      11080      CP      (HL)        ;No imbedded spaces
35DF 2004    11090      JR      NZ,INVMPW
35E1 10FA    11100 CKMPW7 DJNZ     CKMPW5     ;Loop til 8 checked
35E3 AF      11110      XOR      A          ;Set Z = PW good
35E4 C9      11120      RET
                11130 ;
35E5 21DA39  11140 INVMPW LD      HL,INVMPW$   ;Init "Invalid PW
35E8 3E3F    11150      LD      A,63        ;Indicate extended error
35EA B7      11160      OR       A          ;Set NZ condition
35EB C9      11170      RET
                11180 ;
                11190 ;      Brief routine to check a drive for availability
                11200 ;
35EC 210030  11210 CKDRV  LD      HL,HITBUF
35EF         11220      @@TIME      ;P/u the timer pointer
35EF 3E13    00153      LD      A,19
35F1 EF      00154      RST      40
35F2 EB      11230      EX       DE,HL      ;TIME$ to HL
35F3 2B      11240      DEC      HL        ;TIMER$ to HL
35F4 7E      11250      LD      A,(HL)     ;P/u current timer value
35F5 C60F    11260      ADD      A,15      ;Set timeout to 500ms
35F7 57      11270      LD      D,A        ;Save for test later
                11280 ;
                11290 ;      Test for diskette in drive & rotating
                11300 ;
35F8 CD0836  11310 CKDR1 CALL     CKDR6      ;Test index pulse
35FB 20FB    11320      JR      NZ,CKDR1     ;Jump on index
35FD CD0836  11330 CKDR2 CALL     CKDR6      ;Test index pulse
3600 28FB    11340      JR      Z,CKDR2     ;Jump on no index
3602 CD0836  11350 CKDR2A CALL     CKDR6
3605 20FB    11360      JR      NZ,CKDR2A    ;Jump on index
3607 C9      11370      RET
3608 FB      11380 CKDR6 EI
3609 7E      11390      LD      A,(HL)     ;Make sure they're ON
360A 92      11400      SUB      D        ;P/u latest TIMER$ value
360B 2806    11410      JR      Z,CKDR7     ;500ms passed?
360D CD092A  11420      CALL     RSELECT    ;Select & wait not busy
3610 CB4F    11430      BIT      1,A        ;Test index
3612 C9      11440      RET
3613 D1      11450 CKDR7 POP      DE        ;Pop the ret address
3614 F601    11460      OR       1          ;Set "Illegal drive #"
3616 C9      11470      RET              ;With NZ
                11480 ;

```

Format Init Code

```

11490 ;      Temporary storage space for format drive DCT
11500 ;
000A 11510 TMPDCT DS      10
0008 11520 DCTCYL DS      8      ;Default # clys
11530 ;
11540 ;      Config table for single density 5"
11550 ;
3629 11560 TBLDATA EQU     $
3629 EA2A 11570      DW      S5TBL,2409H,3381
      0924 350D
11580 ;
11590 ;      Config table for double density 5"
11600 ;
362F 2A2B 11610      DW      D5TBL,4511H,6506
      1145 6A19
11620 ;
11630 ;      Config table for single density 8"
11640 ;
3635 812B 11650      DW      S8TBL,270FH,5464
      0F27 5815
11660 ;
11670 ;      Config table for double density 8"
11680 ;
363B C62B 11690      DW      D8TBL,491DH,10673
      1D49 B129
11700 ;
11710 ;      Parm error exit
11720 ;
3641 21F439 11730 BADNAM LD      HL,BADNAM$
3644 DD      11740      DB      0DDH
3645 21DA39 11750 BADMPW LD      HL,INVMPW$
3648 DD      11760      DB      0DDH
3649 214737 11770 NOTHARD LD      HL,HARD$
364C C3BC29 11780      JP      EXTERR
364F 3E2C 11790 PRMERR LD      A,44      ;Init Parm ERROR
3651 C3A529 11800      JP      IOERR
11810 ;
11820 ;      Load SYS2 overlay
11830 ;
3654 3E84 11840 GETSYS2 LD      A,84H
3656 EF      11850      RST      28H
11860 ;
3657 20      11870 MPWBUF DB      '      '
      20 20 20 20 20 20 20
11880 PRMTBL$
0080 11890 VAL      EQU      80H
0040 11900 SW      EQU      40H
0020 11910 STR      EQU      20H
0010 11920 SGL      EQU      10H
3660 80      11930      DB      80H
3661 74      11940      DB      SW!STR!SGL!4,'NAME',0
      4E 41 4D 45 00
3666 11950 NRESP      EQU      $-1
3667 EB31 11960      DW      NPARAM+1
3669 73      11970      DB      SW!STR!SGL!3,'MPW',0
      4D 50 57 00
366D 11980 MRESP      EQU      $-1
366E 3B32 11990      DW      MPARAM+1
3670 44      12000      DB      SW!4,'SDEN',0
      53 44 45 4E 00

```

Format Init Code

```

3676 8331      12010      DW      SDPARM+1
3678 44        12020      DB      SW!4,'DDEN',0
      44 44 45 4E 00
367E 8831      12030      DW      DDPARM+1
3680 85        12040      DB      VAL!5,'SIDES',0
      53 49 44 45 53 00
3687 8D31      12050      DW      SIDES+1
3689 93        12060      DB      VAL!SGL!3,'CYL',0
      43 59 4C 00
368E 9231      12070      DW      CPARM+1
3690 84        12080      DB      VAL!4,'STEP',0
      53 54 45 50 00
3696 9731      12090      DW      STEPARM+1
3698 53        12100      DB      SW!SGL!3,'ABS',0
      41 42 53 00
369D 2635      12110      DW      APARM+1
369F 55        12120      DB      SW!SGL!5,'QUERY',0
      51 55 45 52 59 00
36A6 B132      12130      DW      QPARM+1
36A8 46        12140      DB      SW!6,'SYSTEM',0
      53 59 53 54 45 4D 00
36B0 1C26      12150      DW      SYSPRM+1
36B2 94        12160      DB      VAL!SGL!4,'WAIT',0
      57 41 49 54 00
36B8 2C27      12170      DW      WAITPRM+1
36BA 93        12180      DB      VAL!SGL!3,'DIR',0
      44 49 52 00
36BF 1228      12190      DW      DIRPARM+1
36C1 00        12200      NOP
      12210 ;
36C2 46        12220 HELLO$ DB      'FORMAT'
      4F 52 4D 41 54
36C8          12230 *GET      CLIENT:3
      12240 ;CLIENTS/ASM - File to establish sign-on headers
      12250 ;
36C8 20        12260      DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
      2D 20 36 2E 32 2E 30 20
      2D 20 43 6F 70 79 72 69
      67 68 74 20 31 39 38 32
      2F 38 33 2F 38 34 20 62
      79 20 4C 6F 67 69 63 61
      6C
36F2 20        12270      DB      ' Systems, Inc.      ',10
      53 79 73 74 65 6D 73 2C
      20 49 6E 63 2E 20 20 20
      20 20 20 0A
      12280 ;
3707 41        12290      DB      'All Rights Reserved. Licensed 1982/83/84'
      6C 6C 20 52 69 67 68 74
      73 20 52 65 73 65 72 76
      65 64 2E 20 4C 69 63 65
      6E 73 65 64 20 31 39 38
      32 2F 38 33 2F 38 34
372F 20        12300      DB      ' to xxxxxxxxxxxxxxxxx',10,13
      74 6F 20 78 78 78 78 78
      78 78 78 78 78 78 78 78
      78 78 78 78 78 0A 0D
3747 43        12310 HARD$ DB      'Cannot "SYSTEM" a floppy',CR
      61 6E 6E 6F 74 20 22 53

```

Format Init Code

59 53 54 45 4D 22 20 61	
20 66 6C 6F 70 70 79 0D	
3760 49 12320 NOMEM\$ DB	'Insufficient memory for '
6E 73 75 66 66 69 63 69	
65 6E 74 20 6D 65 6D 6F	
72 79 20 66 6F 72 20	
3778 73 12330 DB	'specified format',CR
70 65 63 69 66 69 65 64	
20 66 6F 72 6D 61 74 0D	
3789 57 12340 WHDRV\$ DB	'Which drive is to be used ? ',3
68 69 63 68 20 64 72 69	
76 65 20 69 73 20 74 6F	
20 62 65 20 75 73 65 64	
20 3F 20 03	
37A6 44 12350 DSKNAM\$ DB	'Diskette name ? ',3
69 73 6B 65 74 74 65 20	
6E 61 6D 65 20 3F 20 03	
37B7 4D 12360 MPW\$ DB	'Master password ? ',3
61 73 74 65 72 20 70 61	
73 73 77 6F 72 64 20 3F	
20 03	
37CA 4E 12370 NUMCYL\$ DB	'Number of cylinders ? ',3
75 6D 62 65 72 20 6F 66	
20 63 79 6C 69 6E 64 65	
72 73 20 3F 20 03	
37E1 42 12380 STEP5\$ DB	'Boot strap stepping rate '
6F 6F 74 20 73 74 72 61	
70 20 73 74 65 70 70 69	
6E 67 20 72 61 74 65 20	
37FA 3C 12390 DB	'<6, 12, 20, 30 msecs> ? ',3
36 2C 20 31 32 2C 20 32	
30 2C 20 33 30 20 6D 73	
65 63 73 3E 20 3F 20 03	
3813 42 12400 STEP8\$ DB	'Bootstrap stepping rate '
6F 6F 74 73 74 72 61 70	
20 73 74 65 70 70 69 6E	
67 20 72 61 74 65 20	
382B 3C 12410 DB	'<3, 6, 10, 15/20 msecs> ? ',3
33 2C 20 36 2C 20 31 30	
2C 20 31 35 2F 32 30 20	
6D 73 65 63 73 3E 20 3F	
20 03	
3846 45 12420 SIDES\$ DB	'Enter number of sides <1,2> ? ',3
6E 74 65 72 20 6E 75 6D	
62 65 72 20 6F 66 20 73	
69 64 65 73 20 3C 31 2C	
32 3E 20 3F 20 03	
3865 53 12430 DEN?\$ DB	'Single or Double density <S,D> ? ',3
69 6E 67 6C 65 20 6F 72	
20 44 6F 75 62 6C 65 20	
64 65 6E 73 69 74 79 20	
3C 53 2C 44 3E 20 3F 20	
03	
3887 44 12440 NOTRDY\$ DB	'Drive not ready',CR
72 69 76 65 20 6E 6F 74	
20 72 65 61 64 79 0D	
3897 57 12450 CANTWR\$ DB	'Write protected disk',CR
72 69 74 65 20 70 72 6F	
74 65 63 74 65 64 20 64	

Format Init Code

```

        69 73 6B 0D
38AC 44      12460 NODRV$ DB      'Drive not in system',CR
        72 69 76 65 20 6E 6F 74
        20 69 6E 20 73 79 73 74
        65 6D 0D
38C0 4C      12470 PMTDST$ DB    'Load destination diskette <ENTER>',CR
        6F 61 64 20 64 65 73 74
        69 6E 61 74 69 6F 6E 20
        64 69 73 6B 65 74 74 65
        20 20 3C 45 4E 54 45 52
        3E 0D
38E3 44      12480 HASDAT$ DB    'Disk contains data -- ',3
        69 73 6B 20 63 6F 6E 74
        61 69 6E 73 20 64 61 74
        61 20 2D 2D 20 03
38FA 4E      12490 NOFMT$ DB    'Non-standard format',CR
        6F 6E 2D 73 74 61 6E 64
        61 72 64 20 66 6F 72 6D
        61 74 0D
390E 55      12500 CANTRD$ DB    'Unreadable directory',CR
        6E 72 65 61 64 61 62 6C
        65 20 64 69 72 65 63 74
        6F 72 79 0D
3923 4E      12510 NODIR$ DB    'Non-initialized directory',CR
        6F 6E 2D 69 6E 69 74 69
        61 6C 69 7A 65 64 20 64
        69 72 65 63 74 6F 72 79
        0D
393D 4E      12520 PACKID$ DB    'Name=XXXXXXXX Date=MM/DD/YY',CR
        61 6D 65 3D 58 58 58 58
        58 58 58 58 20 20 44 61
        74 65 3D 4D 4D 2F 44 44
        2F 59 59 0D
395A 20      12530 OLDMPW$ DB    ' Enter its Master Password'
        20 45 6E 74 65 72 20 69
        74 73 20 4D 61 73 74 65
        72 20 50 61 73 73 77 6F
        72 64
3975 20      12540          DB    ' or <BREAK> to abort: ',3
        6F 72 20 3C 42 52 45 41
        4B 3E 20 74 6F 20 61 62
        6F 72 74 3A 20 03
398C 2A      12550 LASTMSG DB    '*** The target drive is a hard disk ***',LF
        2A 2A 20 54 68 65 20 74
        61 72 67 65 74 20 64 72
        69 76 65 20 69 73 20 61
        20 68 61 72 64 20 64 69
        73 6B 20 2A 2A 2A 0A
39B4 41      12560 SURE?$ DB    'Are you sure you want to format it ? ',3
        72 65 20 79 6F 75 20 73
        75 72 65 20 79 6F 75 20
        77 61 6E 74 20 74 6F 20
        66 6F 72 6D 61 74 20 69
        74 20 3F 20 03
39DA 0A      12570 INVMPW$ DB    LF,'Invalid Master Password',LF,CR
        49 6E 76 61 6C 69 64 20
        4D 61 73 74 65 72 20 50
        61 73 73 77 6F 72 64 0A
        0D

```

Format Init Code

```

39F4 49          12580 BADNAM$ DB      'Invalid Disk Name',CR
        6E 76 61 6C 69 64 20 44
        69 73 6B 20 4E 61 6D 65
        0D
3A06 44          12590 PAKNAM$ DB      'DATADISK '
        41 54 41 44 49 53 4B
3A0E 50          12600 PAKMPW$ DB      'PASSWORD '
        41 53 53 57 4F 52 44
        08520 ;
3A16            08530          SUBTTL  <>
3100            08540          END    FORMAT

```


@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
AFLOP	27D3 APARM	3525 BADMPW	3645
BADNAM	3641 BADNAM\$	39F4 BFMT1	26A4
BFMT2	26A7 BFMT3	26CF BFMT4	26D5
BFMT5	26FE BGNFMT	2698 BGNVER	2735
BOOT	2F00 BOOTDIR	2A9D BOOTST\$	2600
BREAK	29B9 BVER1	2747 BVER10	27B2
BVER3	2767 BVER4	2768 BVER5	2771
BVER8	278A BVER9	27A9 CALC1	2823
CALC2	2836 CALC3	283A CALCDIR	2806
CALCGPC	33E1 CANTRD\$	390E CANTWR\$	3897
CGPC1	33F4 CGPC2	33FF CGPC3	3407
CHGDEN	32F5 CKDR1	35F8 CKDR2	35FD
CKDR2A	3602 CKDR6	3608 CKDR7	3613
CKDRV	35EC CKMPW	35B5 CKMPW0	35BC
CKMPW1	35C3 CKMPW2	35D1 CKMPW3	35D9
CKMPW5	35DD CKMPW7	35E1 CKNAME	322B
CKWAIT	2724 CODE1	265F CODE1A	2661
CODF1	2669 CODF1A	266C CODF2	2670
CODF2A	2677 CODF3	267A CODF4	2682
CODF5	2657 CODRET	2665 CORE\$	2F00
CPARM	3191 CR	000D CRT3	3C00
CRT4	F800 CVB1	3584 CVBIN	3582
CVD1	2A52 CVD2	2A59 CVDEC	2A50
CYLGRN	2951 D5TBL	2B2A D8TBL	2BC6
DATADSK\$	4329 DCTCYL	3621 DCTLPI	313A
DDPARM	3187 DEN?\$	3865 DFTMPW	3245
DFTNAM	31F5 DIRASC\$	2CC6 DIRCYL\$	2CA1
DIRDIR	2ABD DIRPARM	2811 DIRSET	2821
DRVNOP	29F6 DSKNAM	3218 DSKNAM\$	37A6
DSPCYL	2A89 ERREXIT	29C5 EXIT	29C8
EXIT2	29DF EXIT3	29ED EXIT4	29F4
EXTERR	29BC FILLBLK	35AE FMT1	3144
FMT2	316D FMT2A	3175 FMT2B	31E4
FMTABT	29B9 FMTABT\$	2D24 FMTCAO\$	2D0E
FMTCYL\$	2C39 FMTDAT	2639 FMTDRV	2A2A
FMTG\$	2C6E FMTHD	2A13 FMTTBL	2608
FORMAT	3100 FORMATA	3109 GATBUF	2E00
GDDEN1	32D7 GENSYS	2840 GET3	2A5D
GET8	2A65 GET8A	2A6B GETDAT	3234
GETDEN	32B8 GETSYS2	3654 GETUC	2A77
GETUC1	2A82 GOFMT	2601 GOTSTEP	33DD
GSDEN1	32E7 GSYS1	28F2 GSYS2	2981
GSYS3	298B HARD\$	3747 HASDAT\$	38E3
HASHMPW	35B9 HELLO\$	36C2 HITBUF	3000
HRDRV	27C1 HRDRV1	27E5 INITDEN	328C
INMPW	3595 INVMPW	35E5 INVMPW\$	39DA
IOERR	29A5 IPLSYS\$	2CC9 LASTMSG	398C
LF	000A LSIID	28DB MOVFREE	2794
MOVMPW	324D MOVMPW1	325C MOVNAM	31FD
MOVNAM1	3211 MPARM	323A MPW	3263
MPW\$	37B7 MPWBUF	3657 MRESP	366D
NOCYL\$	2C7C NODIR	281E NODIR\$	3923
NODRV\$	38AC NOFMT\$	38FA NOMEM\$	3760
NOFMT\$	2D35 NOTHARD	3649 NOTRDY\$	3887
NPARM	31EA NRESP	3666 NUMCYL\$	37CA
OLDMPW	3508 OLDMPW\$	395A PACKID\$	393D

PAKMPW\$	3A0E	PAKNAM\$	3A06	PASSWORD	42E0
PCYL1	3330	PCYL2	333C	PCYL3	3341
PCYL4	334F	PFMT1	3432	PFMT2	3479
PFMT3	348F	PFMT4	34CD	PFMT5	3522
PFMT6	3545	PMTCYL	3329	PMTDST	3412
PMTDST\$	38C0	PMTS1	331E	PMTSIDE	32F8
PMTSTEP	3363	PMTSYS\$	2CEF	PRMERR	364F
PRMMRG	32AB	PRMTBL\$	3660	PRSMPW	326B
PSTEP1	33D6	QPARM	32B0	RDSEC	2A22
RESTOR	29FF	RETCOD	29C9	RLS	0062
RSELECT	2A09	S5TBL	2AEA	S8TBL	2B81
SAFESP	4400	SDPARM	3182	SECCYL	2AE8
SECSKEW	26A1	SECTRK	2AE9	SELECT	29FA
SETDDEN	32DE	SETSDEN	32EE	SETSTD	32A4
SETUP	3559	SGL	0010	SIDES	318C
SIDES\$	3846	SPSAV	29D8	STAR\$	2C69
STEP5	3382	STEP5\$	37E1	STEP8	33AC
STEP8\$	3813	STEPARM	3196	STEPDFT	2AE7
STEPIIN	2A04	STOP	4326	STR	0020
STRLEN	0017	SURE?\$	39B4	SW	0040
SYSRCT	2ADD	SYSPRM	261B	TBLDATA	3629
TMPDCT	3617	TRKSKEW	2702	TSTSID	3323
VAL	0080	VERCYL\$	2C51	VERS1	2A3C
VERSEC	2A27	VERSKW	2764	VERSYS	2A31
WAITPRM	272B	WHDRV	3151	WHDRV\$	3789
WRCYL	2A0E	WRDIR	2A41	WRDIR1	2A44
WRGAT1	28DF	WRSEC	2A18	WRSYS	2A1D
@@ABORT	BE7D	@@ADTSK	BF10	@@BANK	C428
@@BKSP	C108	@@BREAK	C43E	@@CHNIO	BE68
@@CKBRKC	C48C	@@CKDRV	BF64	@@CKEOF	C11D
@@CKTSK	BEFB	@@CLOSE	C0F3	@@CLS	C476
@@CMNDI	BEA7	@@CMNDR	BEBC	@@CTL	BCCC
@@DATE	BE3E	@@DCSTAT	BFA3	@@DEBUG	BEE6
@@DECHX	C3A8	@@DIRRD	C315	@@DIRWR	C32A
@@DIV16	C393	@@DIV8	C37E	@@DODIR	BF79
@@DSP	BC90	@@DSPLY	BD30	@@ERROR	BED1
@@EXIT	BE92	@@FEXT	C282	@@FLAGS	C412
@@FNAME	C297	@@FSPEC	C26D	@@GATRD	C300
@@GATWR	C33F	@@GET	BCA4	@@GTDCB	C2C1
@@GTDCT	C2AC	@@GTMOD	C2D6	@@HDFMT	C04B
@@HEX16	C3E7	@@HEX8	C3D2	@@HEXDEC	C3BD
@@HIGH\$	C3FC	@@INIT	C0C9	@@KBD	BD08
@@KEY	BC7C	@@KEYIN	BD1C	@@KLTsk	BF4F
@@LOAD	C243	@@LOC	C132	@@LOF	C147
@@LOGGER	BD67	@@LOGOT	BD7C	@@MSG	BDB3
@@MUL16	C369	@@MUL8	C354	@@OPEN	C0DE
@@PARAM	BE29	@@PAUSE	BE14	@@PEOF	C15C
@@POSN	C171	@@PRINT	BDC8	@@PRT	BCE0
@@PUT	BCB8	@@RAMDIR	BF8E	@@RDSEC	C021
@@RDSSC	C2EB	@@READ	C186	@@REMOV	C0B4
@@RENAM	C09F	@@REW	C19B	@@RMTSK	BF25
@@RPTSK	BF3A	@@RREAD	C1B0	@@RSLCT	C00C
@@RSTOR	BFC0	@@RUN	C258	@@RWRTT	C1C5
@@SEEK	BFF7	@@SEEKSC	C1DA	@@SKIP	C1EF
@@SLCT	BFB8	@@STEPI	BFE2	@@TIME	BE53
@@VDCtl	BDFF	@@VER	C204	@@VRSEC	C036
@@WEOF	C219	@@WHERE	BCF4	@@WRITE	C22E
@@WRSEC	C060	@@WRSSC	C075	@@WRTRK	C08A

00000 Total errors

NOTES:

NOTES:

FORMS/FLT - Printer output formatting filter

The Forms filter allows formatting of data sent to the *PR device. It is installed with the SET and FILTER Library commands. Its parameters are adjusted with the FORMS Library command.

```

00100 ;FORMS/ASM - Printer Formatting Filter
0000 00110 TITLE <FORMS/FLT - LS-DOS 6.2>
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
00150 ;
0000 00160 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00170 *GET COPYCOM:3 ;Copyright message
03920 ; COPYCOM - File for Copyright COMment block
03930 ;
0000 03940 COM '<*(C) 1982,83,84 by LSI*>'
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 BEGIN
2400 00220 @@CKBRKC ;Check for break
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,BEGINA ;Continue if no Break
2405 21FFFF 00240 LD HL,-1
2408 C9 00250 RET ; else abort
00260 ;
2409 D5 00270 BEGINA PUSH DE ;Save DCB address
240A DDE1 00280 POP IX ; in index reg
240C ED534B26 00290 LD (PFDCB),DE ; and in filter header
2410 00300 @@DSPLY HELLO$ ;Welcome the user
00003 IFEQ 01H,1
2410 212525 00004 LD HL,HELLO$
00005 ENDIF
2413 3E0A 00006 LD A,10
2415 EF 00007 RST 40
00310 ;
00320 ; Check if entry from SET command
00330 ;
2416 00340 @@FLAGS ;IY => flag table base
2416 3E65 00008 LD A,101
2418 EF 00009 RST 40
2419 FDCB025E 00350 BIT 3,(IY+'C'-'A') ;System request?
241D CA0525 00360 JP Z,VIASET ;Quit if not
00370 ;
00380 ; Check if filter is already resident
00390 ;
2420 112125 00400 LD DE,FF$ ;Check if filter is
2423 00410 @@GTMOD ; already resident
2423 3E53 00010 LD A,83
2425 EF 00011 RST 40
2426 EB 00420 EX DE,HL ;Put DCB ptr to HL
2427 2023 00430 JR NZ,NOTRES ;Go if not
00440 ;
00450 ; Make sure that the new DCB is same as the old
00460 ;
2429 ED4B4B26 00470 LD BC,(PFDCB) ;Replace DCB pointer
242D 79 00480 LD A,C ; with new one
242E 4E 00490 LD C,(HL) ;P/u DCB pointer LSB
242F 00 00500 NOP
2430 23 00510 INC HL
2431 78 00520 LD A,B
2432 46 00530 LD B,(HL) ;P/u DCB pointer MSB

```

The Source	UTILITY Files	FORMS/FLT - LS-DOS 6.2	Page 00002
2433 210600	00540	LD HL,6 ;Get old DCB name &	
2436 09	00550	ADD HL,BC ; stuff into error	
2437 7E	00560	LD A,(HL) ; message in case	
2438 2C	00570	INC L ; a different DCB	
2439 66	00580	LD H,(HL) ; is referenced	
243A 6F	00590	LD L,A	
243B 22FB25	00600	LD (DCBNAM\$),HL ;Stuff message with spec	
243E B4	00610	OR H	
243F 2876	00620	JR Z,ISRES	
2441 2A4B26	00630	LD HL,(PFDCB) ;P/u DCB existing DCB	
2444 B7	00640	OR A ; pointer	
2445 ED42	00650	SBC HL,BC ;Same DCB pointer?	
2447 C20925	00660	JP NZ,DCBERR ;Can't install if diff	
244A 186B	00670	JR ISRES	
	00680 ;		
	00690 ;	Module is not resident	
	00700 ;		
244C 114B49	00710	NOTRES LD DE,'IK'	
244F	00720	@@GTDCB ;Locate low memory ptr	
244F 3E52	00012	LD A,82	
2451 EF	00013	RST 40	
2452 C21725	00730	JP NZ,IOERR ;Quit if not found	
2455 2D	00740	DEC L	
2456 56	00750	LD D,(HL) ;P/u pointer to	
2457 2D	00760	DEC L ; start of free	
2458 5E	00770	LD E,(HL) ; low core	
2459 ED53A724	00780	LD (LCPTR+1),DE ;Save loc for later	
245D E5	00790	PUSH HL ;Save low core ptr	
245E 210101	00800	LD HL,PFEND-PFFLT	
2461 19	00810	ADD HL,DE ;Start + driver length	
2462 E5	00820	PUSH HL	
2463 2B	00830	DEC HL ;Point to last byte	
2464 22DD24	00840	LD (SVEND+1),HL	
2467 010013	00850	LD BC,1300H ;Max addr + 1	
246A AF	00860	XOR A	
246B ED42	00870	SBC HL,BC	
246D D1	00880	POP DE ;Rcvr new lc	
246E E1	00890	POP HL ;Rcvr low core ptr	
246F 382F	00900	JR C,PUTLOW ;If room, put low	
	00910 ;		
	00920 ;	Check if high memory available	
	00930 ;		
2471	00940	@@FLAGS	
2471 3E65	00014	LD A,101	
2473 EF	00015	RST 40	
2474 FDCB0246	00950	BIT 0,(IY+'C'-'A') ;Memory frozen?	
2478 C20D25	00960	JP NZ,NOROOM ;"No memory..."	
247B 210000	00970	LD HL,0 ;Get HIGH\$	
247E 45	00980	LD B,L	
247F	00990	@@HIGH\$	
247F 3E64	00016	LD A,100	
2481 EF	00017	RST 40	
2482 22DD24	01000	LD (SVEND+1),HL ;Save for relocater	
2485 5D	01010	LD E,L ;Xfer new last	
2486 54	01020	LD D,H ; to reg DE	
2487 AF	01030	XOR A ;Calc new start	
2488 010101	01040	LD BC,PFEND-PFFLT ;BC = filter len	
248B ED42	01050	SBC HL,BC	
248D 0600	01060	LD B,0	
248F	01070	@@HIGH\$;Set new HIGH\$	
248F 3E64	00018	LD A,100	


```

2491 EF      00019      RST      40
2492 23      01080      INC      HL                      ;Point to new start
2493 EB      01090      EX       DE,HL
2494 D5      01100      PUSH     DE
2495 CDD624   01110      CALL     RELO                      ;Relocate internal references
2498 D1      01120      POP      DE
2499 3EFF     01130      LD       A,0FFH
249B 32C824   01140      LD       (HGHL),A                ;Flag to notify user
249E 1809     01150      JR       MOVMOD                    ; himem used
                01160 ;
                01170 ;      Room in low core - move driver low
                01180 ;
24A0 73      01190 PUTLOW LD      (HL),E                ;Stuff low core ptr
24A1 2C      01200      INC      L                      ; with new low
24A2 72      01210      LD       (HL),D
24A3 CDD624   01220      CALL     RELO                      ;Relocate vectors
24A6 110000   01230 LCPTR  LD      DE,$-$                ;Low core pointer
                01240 ;
                01250 ;      Move module to memory
                01260 ;
24A9 D5      01270 MOVMOD PUSH     DE                      ;Save start
24AA 214326   01280      LD       HL,PFFLT
24AD 010101   01290      LD       BC,PFEND-PFFLT          ;Calc driver length
24B0 EDB0     01300      LDIR
24B2 D1      01310      POP      DE                      ;Pop filter start
24B3 FDCB03EE 01320      SET      5,(IY+'D'-'A')          ;Set PF in DFLAG$
                01330 ;
24B7 21FE25   01340 ISRES LD      HL,PFAC$                ;Init "FORMS installed
24BA DD360047 01350      LD      (IX),40H!7              ;Init DCB type to "C/P/G"
24BE DD7301   01360      LD      (IX+1),E                ; & filter & stuff the
24C1 DD7202   01370      LD      (IX+2),D                ; filter address
24C4          01380 @@LOGOT                                ;Display installation
                00020 IFEQ     00H,1
                00021 LD      HL,
                00022 ENDIF
24C4 3E0C     00023 LD      A,12
24C6 EF      00024 RST      40
24C7 3E00     01390 LD      A,$-$
24C8          01400 HGHLG EQU     $-1                      ;Flag filter went high
24C9 B7      01410 OR       A                      ;Skip if not set
24CA 2806     01420 JR       Z,NTHGH
24CC 211B26   01430 LD      HL,HMEM$                ; else show "Went in himem
24CF          01440 @@LOGOT
                00025 IFEQ     00H,1
                00026 LD      HL,
                00027 ENDIF
24CF 3E0C     00028 LD      A,12
24D1 EF      00029 RST      40
24D2 210000   01450 NTHGH LD      HL,0                ;No error
24D5 C9      01460 RET                                ;Done, back to user
                01470 ;
                01480 ;      Relocate internal references in driver
                01490 ;
24D6 DDE5     01500 RELO   PUSH     IX
24D8 DD214427 01510      LD      IX,RELTAB                ;Point to relocation tbl
24DC 210000   01520 SVEND LD      HL,$-$                ;Find distance to move
24DF 224526   01530      LD      (PFFLT+2),HL          ;Set last byte used
24E2 114327   01540      LD      DE,PFEND-1
24E5 B7      01550 OR       A                      ;Clear carry flag
24E6 ED52     01560 SBC      HL,DE
24E8 44      01570 LD      B,H                      ;Move to BC

```

```

24E9 4D      01580      LD      C,L
24EA 3E0E    01590      LD      A,TABLEN      ;Get table length
24EC DD6E00  01600 RLOOP  LD      L,(IX)      ;Get address to change
24EF DD6601  01610      LD      H,(IX+1)
24F2 5E      01620      LD      E,(HL)      ;P/U address
24F3 23      01630      INC     HL
24F4 56      01640      LD      D,(HL)
24F5 EB      01650      EX      DE,HL      ;Offset it
24F6 09      01660      ADD     HL,BC
24F7 EB      01670      EX      DE,HL
24F8 72      01680      LD      (HL),D      ;Put it back
24F9 2B      01690      DEC     HL
24FA 73      01700      LD      (HL),E
24FB DD23    01710      INC     IX
24FD DD23    01720      INC     IX
24FF 3D      01730      DEC     A
2500 20EA    01740      JR      NZ,RLOOP      ;Loop till done
2502 DDE1    01750      POP     IX
2504 C9      01760      RET
          01770 ;
          01780 ;      Error exits
          01790 ;
2505 21B025  01800 VIASET LD      HL,VIASET$      ;"Install with Set
2508 DD      01810      DB      0DDH
2509 21DF25  01820 DCBERR LD      HL,DCBERR$      ;"Filter in use
250C DD      01830      DB      0DDH
250D 21C525  01840 NOROOM LD      HL,NOROOM$      ;"Memory frozen
2510         01850      @@LOGOT      ;Show the error
          00030      IFEQ    00H,1
          00031      LD      HL,
          00032      ENDIF
2510 3E0C    00033      LD      A,12
2512 EF      00034      RST     40
2513 21FFFF  01860      LD      HL,-1      ;Set abort code
2516 C9      01870      RET
          01880 ;
2517 6F      01890 IOERR  LD      L,A      ;Error # to HL
2518 2600    01900      LD      H,0
251A F6C0    01910      OR      0C0H      ;Abbrev, return
251C 4F      01920      LD      C,A      ;Error code to C
251D         01930      @@ERROR      ; for error display
251D 3E1A    00035      LD      A,26
251F EF      00036      RST     40
2520 C9      01940      RET
          01950 ;
          01960 ;      Messages & Data tables
          01970 ;
2521 24      01980 FF$    DB      '$FF',3
          46 46 03
2525 46      01990 HELLO$ DB      'FORMS Filter'
          4F 52 4D 53 20 46 69 6C
          74 65 72
2531         02000 *GET     CLIENT:3
          03950 ;CLIENTS/ASM - File to establish sign-on headers
          03960 ;
2531 20      03970      DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
          2D 20 36 2E 32 2E 30 20
          2D 20 43 6F 70 79 72 69
          67 68 74 20 31 39 38 32
          2F 38 33 2F 38 34 20 62
          79 20 4C 6F 67 69 63 61

```

```

        6C
255B 20      03980      DB      ' Systems, Inc.      ',10
      53 79 73 74 65 6D 73 2C
      20 49 6E 63 2E 20 20 20
      20 20 20 0A
      03990 ;
2570 41      04000      DB      'All Rights Reserved. Licensed 1982/83/84'
      6C 6C 20 52 69 67 68 74
      73 20 52 65 73 65 72 76
      65 64 2E 20 4C 69 63 65
      6E 73 65 64 20 31 39 38
      32 2F 38 33 2F 38 34
2598 20      04010      DB      ' to xxxxxxxxxxxxxxxxxxxx',10,13
      74 6F 20 78 78 78 78 78
      78 78 78 78 78 78 78 78
      78 78 78 78 78 0A 0D
      02010 ;
25B0 4D      02020 VIASET$ DB      'Must install via SET',CR
      75 73 74 20 69 6E 73 74
      61 6C 6C 20 76 69 61 20
      53 45 54 0D
25C5 4E      02030 NOROOM$ DB      'No memory space available',CR
      6F 20 6D 65 6D 6F 72 79
      20 73 70 61 63 65 20 61
      76 61 69 6C 61 62 6C 65
      0D
25DF 46      02040 DCBERR$ DB      'Filter already attached to *xx',CR
      69 6C 74 65 72 20 61 6C
      72 65 61 64 79 20 61 74
      74 61 63 68 65 64 20 74
      6F 20 2A 78 78 0D
25FB      02050 DCBNAM$ EQU      $-3
25FE 46      02060 PFACT$ DB      'Forms filter is now resident',CR
      6F 72 6D 73 20 66 69 6C
      74 65 72 20 69 73 20 6E
      6F 77 20 72 65 73 69 64
      65 6E 74 0D
261B 0A      02070 HMEM$ DB      LF,'Note: filter installed in high memory.',CR
      4E 6F 74 65 3A 20 66 69
      6C 74 65 72 20 69 6E 73
      74 61 6C 6C 65 64 20 69
      6E 20 68 69 67 68 20 6D
      65 6D 6F 72 79 2E 0D
      02080 ;
      02090 ;
      02100 ;
      02110 ;
      02120 ;
      02130 ;
      02140 ;
      02150 ;
      02160 *MOD
0003      02170 PFBIT EQU      3      ;Position in DFLAG
0000      02180 SPLBIT EQU      0      ;Position in DFLAG
000A      02190 LF EQU      10
000D      02200 CR EQU      13
      02210 ;
2643 1814      02220 PFFLT JR      PFBGN      ;Branch around header
2645 4327      02230      DW      PFEND-1      ;Last byte used
2647 03      02240      DB      3,'$FF'      ;Name length/name
      24 46 46

```

The Source	UTILITY Files	FORMS/FLT - LS-DOS 6.2	Page 00006
264B 0000	02250 PFDCB	DW \$-\$;Link to DCB
264D 0000	02260	DW 0	
	02270 ;		
	02280 ;	Filter data area	
	02290 ;		
264F 0000	02300 PFDATA\$	EQU \$	
0000	02310 PMAX	EQU \$-PFDATA\$	
264F 42	02320	DB 66	;Page size (max lines per page)
0001	02330 LCOUNT	EQU \$-PFDATA\$	
2650 00	02340	DB 0	;Line counter
0002	02350 LMAX	EQU \$-PFDATA\$	
2651 42	02360	DB 66	;Max lines to print
0003	02370 CCOUNT	EQU \$-PFDATA\$	
2652 00	02380	DB 0	;Chars per line printed
0004	02390 XL1	EQU \$-PFDATA\$	
2653 00	02400	DB 0	;Translate from
0005	02410 XL2	EQU \$-PFDATA\$	
2654 00	02420	DB 0	;Translate to
0006	02430 INDENT	EQU \$-PFDATA\$	
2655 00	02440	DB 0	;Indent after line wraparound
0007	02450 ADDLF	EQU \$-PFDATA\$	
2656 04	02460	DB 4	;Bit-0, LF after CR; bit-1=FF
	02470		;Bit-2, TAB expand (1)
0008	02480 CMAX	EQU \$-PFDATA\$	
2657 00	02490	DB 0	;Max CPL before wraparound
0009	02500 MARGIN	EQU \$-PFDATA\$	
2658 00	02510	DB 0	;Left hand margin
	02520 ;		
	02530 ;	Start of filter	
	02540 ;		
2659 281A	02550 PFBGN	JR Z,FFENTRY	;Go if @PUT
265B 11	02560	DB 011H	;Ignore next inst if not
265C 0602	02570 PFPUT	LD B,2	;Init for @PUT
265E DDE5	02580	PUSH IX	
2660 DD2A4B26	02590	LD IX,(PFDCB)	;Grab the DCB vector
2662	02600 RX01	EQU \$-2	
2664	02610	@@CHNIO	; & chain to it
2664 3E14	00037	LD A,20	
2666 EF	00038	RST 40	
2667 DDE1	02620	POP IX	
2669 C9	02630	RET	
	02640 ;		
	02650 ;	Peform the tab function	
	02660 ;		
266A DD7E03	02670 DOTAB	LD A,(IX+CCOUNT)	;How many spaces to
266D E607	02680	AND 7	; next tab stop?
266F D608	02690	SUB 8	
2671 ED44	02700	NEG	
2673 1867	02710	JR @INDENT	;Space over to it
	02720 ;		
	02730 ;	Filter code	
	02740 ;		
2675 DD214F26	02750 FFENTRY	LD IX,PFDATA\$;Base register
2677	02760 RX02	EQU \$-2	
	02770 ;		
2679 DD7E04	02780 CKXLAT	LD A,(IX+XL1)	;Get xlate in
267C B9	02790	CP C	;Translate this char?
267D 2004	02800	JR NZ,CONT	;Go if not xlated char
267F DD7E05	02810	LD A,(IX+XL2)	;Xlated to this
2682 4F	02820	LD C,A	
2683 79	02830 CONT	LD A,C	;P/u char to test

```

2684 FE0C        02840            CP        0CH            ;Form feed?
2686 CA1B27     02850            JP        Z,DOTOF
2687            02860        RX14    EQU        $-2
2689 FE06        02870            CP        6            ;SET TOF?
268B CA3E27     02880            JP        Z,SETTOF
268C            02890        RX03    EQU        $-2
268E FE0D        02900            CP        CR            ;CR?
2690 287A        02910            JR        Z,DOCRLF
2692 FE0A        02920            CP        LF            ;LF?
2694 2876        02930            JR        Z,DOCRLF
2696 DD7E09     02940            LD        A,(IX+MARGIN) ;Left margin to do?
2699 B7          02950            OR        A
269A 280B        02960            JR        Z,NOMARG        ;Go if not
269C DD3403     02970            INC        (IX+CCOUNT)     ;Check current char count
269F DD3503     02980            DEC        (IX+CCOUNT)     ;If at newline,
26A2 C5          02990            PUSH      BC
26A3 CCDC26     03000            CALL      Z,@INDENT        ; need a margin now
26A4            03010        RX13    EQU        $-2
26A6 C1          03020            POP       BC
26A7 79          03030        NOMARG   LD        A,C            ;P/u character again
26A8 DDCB0756   03040            BIT        2,(IX+ADDLF)    ;Expand tabs?
26AC 2804        03050            JR        Z,CONTA
26AE FE09        03060            CP        9            ;Tab?
26B0 28B8        03070            JR        Z,DOTAB
26B2 FE20        03080        CONTA   CP        20H            ;Other control code?
26B4 38A6        03090            JR        C,PFPUT        ;Pass on unchanged if so
                 03100 ;
                 03110 ;        Got a character to output
                 03120 ;
26B6 C5          03130        PUTCHAR   PUSH      BC            ;Save character
26B7 CDC026     03140            CALL      SETUP            ;Setup for next char
26B8            03150        RX12    EQU        $-2
26BA C1          03160            POP       BC
26BB C0          03170            RET       NZ            ;Quit on error
26BC CC5C26     03180            CALL      Z,PFPUT        ;Now put the char
26BD            03190        RX04    EQU        $-2
26BF C9          03200            RET
                 03210 ;
                 03220 ;        Do the end of line check
                 03230 ;
26C0 DD3403     03240        SETUP   INC        (IX+CCOUNT)     ;Inc char counter
26C3 DD7E08     03250            LD        A,(IX+CMAX)     ;Wraparound needed?
26C6 A7          03260            AND       A
26C7 C8          03270            RET       Z            ;Quit if feature is off
26C8 DDBE03     03280            CP        (IX+CCOUNT)
26CB 3075        03290            JR        NC,EXITZ        ;Done if not needed
26CD CD0C27     03300            CALL      DOCRLF        ;Do carriage return
26CE            03310        RX05    EQU        $-2
26D0 C0          03320            RET       NZ
26D1 DD3403     03330            INC        (IX+CCOUNT)     ;Adjust char counter
                 03340 ;
                 03350 ;        Check on indent needed
                 03360 ;
26D4 DD7E06     03370            LD        A,(IX+INDENT)    ;P/u indent
26D7 DD8609     03380            ADD       A,(IX+MARGIN)    ;Add in the MARGIN
26DA B7          03390            OR        A
26DB C8          03400            RET       Z            ;Done if none
26DC C5          03410        @INDENT   PUSH      BC            ;In case of recursive
26DD 47          03420            LD        B,A            ; calls
26DE 0E20        03430            LD        C,' '          ;Print spaces
26E0 C5          03440        SPACES   PUSH      BC            ;Save counter

```

The Source	UTILITY Files	FORMS/FLT - LS-DOS 6.2	Page 00008
26E1 AF	03450	XOR A	
26E2 CDB626	03460	CALL PUTCHAR	;Put the character
26E3	03470 RX06	EQU \$-2	
26E5 C1	03480	POP BC	;Recover counter
26E6 2002	03490	JR NZ,\$+4	;Exit on PUT error
26E8 10F6	03500	DJNZ SPACES	
26EA C1	03510	POP BC	
26EB C9	03520	RET	
26EC DDCB0746	03530 LINFEED	BIT 0,(IX+ADDLF)	
26F0 2808	03540	JR Z,DOWN1	;Go if hardware auto-LF
26F2 0E0D	03550	LD C,CR	;Else do CR and LF
26F4 CD5C26	03560	CALL PFPUT	
26F5	03570 RX11	EQU \$-2	
26F7 C0	03580	RET NZ	
26F8 1808	03590	JR DOWNLF	
26FA DD7E03	03600 DOWN1	LD A,(IX+CCOUNT)	
26FD A7	03610	AND A	;Line empty?
26FE 0E0D	03620	LD C,CR	;Do CR if not
2700 2002	03630	JR NZ,DOWNCR	
2702 0E0A	03640 DOWNLF	LD C,LF	;Do LF if so
2704 CD5C26	03650 DOWNCR	CALL PFPUT	
2705	03660 RX07	EQU \$-2	
2707 DD360300	03670	LD (IX+CCOUNT),0	;Starting new line
270B C9	03680	RET	
	03690 ;		
270C CDEC26	03700 DOCRLF	CALL LINFEED	;CRLF & check if page end
270D	03710 RX08	EQU \$-2	
270F C0	03720	RET NZ	
	03730 ;		
2710 DD3401	03740	INC (IX+LCOUNT)	
2713 DD7E01	03750	LD A,(IX+LCOUNT)	;Time to do form feed?
2716 DDBE02	03760	CP (IX+LMAX)	
2719 3827	03770	JR C,EXITZ	;Return if not
	03780 ;		
271B DD7E00	03790 DOTOF	LD A,(IX+PMAX)	;How many lines to feed?
271E DD9601	03800	SUB (IX+LCOUNT)	
2721 281B	03810	JR Z,SETTOF	;Skip if zero
2723 C5	03820	PUSH BC	;In case called by DOTAB
2724 47	03830	LD B,A	
2725 DDCB074E	03840	BIT 1,(IX+ADDLF)	;Hardware form feed?
2729 2807	03850	JR Z,SOFTFF	;Go if not
272B 0E0C	03860	LD C,0CH	; else load up TOF char
272D CD5C26	03870	CALL PFPUT	; and send it
272E	03880 RX09	EQU \$-2	
2730 180B	03890	JR FFEXIT	
2732 C5	03900 SOFTFF	PUSH BC	
2733 CDEC26	03910	CALL LINFEED	;Do LF's
2734	03920 RX10	EQU \$-2	
2736 C1	03930	POP BC	
2737 2802	03940	JR Z,CHRGONE	;This linefeed sent OK
2739 C1	03950	POP BC	; else clean stack
273A C9	03960	RET	; and return error
273B 10F5	03970 CHRGONE	DJNZ SOFTFF	
273D C1	03980 FFEXIT	POP BC	
	03990 ;		
	04000 ;	Set the top-of-form	
	04010 ;		
273E DD360100	04020 SETTOF	LD (IX+LCOUNT),0	;Reset line counter
2742 BF	04030 EXITZ	CP A	
2743 C9	04040	RET	
	04050 ;		

```

2744      04060 PFEND  EQU  $
          04070 ;
2744 6226      04080 RELTAB  DW  RX01,RX02,RX03,RX04,RX05,RX06,RX07,RX08
          7726 8C26 BD26 CE26 E326 0527 0D27
2754 2E27      04090          DW  RX09,RX10,RX11,RX12,RX13,RX14
          3427 F526 B826 A426 8726
000E      04100 TABLEN EQU  $-RELTAB/2
          04110 ;
2400      04120          END  BEGIN

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @INDENT	26DC @MOD2	0000
@MOD4	FFFF ADDLF	0007 BEGIN	2400
BEGINA	2409 CCOUNT	0003 CHRGONE	273B
CK XLAT	2679 CMAX	0008 CONT	2683
CONTA	26B2 CR	000D DCBERR	2509
DCBERR\$	25DF DCBNAM\$	25FB DOCRFL	270C
DOTAB	266A DOTOF	271B DOWN1	26FA
DOWNCR	2704 DOWNLF	2702 EXITZ	2742
FF\$	2521 FFENTRY	2675 FFEXIT	273D
HELLO\$	2525 HGHLFG	24C8 HMEM\$	261B
INDENT	0006 IOERR	2517 ISRES	24B7
LCOUNT	0001 LCPTR	24A6 LF	000A
LINFEED	26EC LMAX	0002 MARGIN	0009
MOVMOD	24A9 NOMARG	26A7 NOROOM	250D
NOROOM\$	25C5 NOTRES	244C NTHGH	24D2
PFACT\$	25FE PFBGN	2659 PFBIT	0003
PFDATA\$	264F PFDCB	264B PFEND	2744
PFFLT	2643 PFPUT	265C PMAX	0000
PUTCHAR	26B6 PUTLOW	24A0 RELO	24D6
RELTAB	2744 RLOOP	24EC RX01	2662
RX02	2677 RX03	268C RX04	26BD
RX05	26CE RX06	26E3 RX07	2705
RX08	270D RX09	272E RX10	2734
RX11	26F5 RX12	26B8 RX13	26A4
RX14	2687 SETTOF	273E SETUP	26C0
SOFTFF	2732 SPACES	26E0 SPLBIT	0000
SVEND	24DC TABLEN	000E VIASET	2505
VIASET\$	25B0 XL1	0004 XL2	0005
@@ABORT	8FAE @@ADTSK	9041 @@BANK	9559
@@BKSP	9239 @@BREAK	956F @@CHNIO	8F99
@@CKBRKC	95BD @@CKDRV	9095 @@CKEOF	924E
@@CKTSK	902C @@CLOSE	9224 @@CLS	95A7
@@CMNDI	8FD8 @@CMNDR	8FED @@CTL	8DFD
@@DATE	8F6F @@DCSTAT	90D4 @@DEBUG	9017
@@DECHEX	94D9 @@DIRRD	9446 @@DIRWR	945B
@@DIV16	94C4 @@DIV8	94AF @@DODIR	90AA
@@DSP	8DC1 @@DSPLY	8E61 @@ERROR	9002
@@EXIT	8FC3 @@FEXT	93B3 @@FLAGS	9543
@@FNAME	93C8 @@FSPEC	939E @@GATRD	9431
@@GATWR	9470 @@GET	8DD5 @@GTDCB	93F2
@@GTDC	93DD @@GTMOD	9407 @@HDFMT	917C
@@HEX16	9518 @@HEX8	9503 @@HEXDEC	94EE
@@HIGH\$	952D @@INIT	91FA @@KBD	8E39
@@KEY	8DAD @@KEYIN	8E4D @@KLTSK	9080
@@LOAD	9374 @@LOC	9263 @@LOF	9278
@@LOGGER	8E98 @@LOGOT	8EAD @@MSG	8EE4
@@MUL16	949A @@MUL8	9485 @@OPEN	920F
@@PARAM	8F5A @@PAUSE	8F45 @@PEOF	928D
@@POSN	92A2 @@PRINT	8EF9 @@PRT	8E11
@@PUT	8DE9 @@RAMDIR	90BF @@RDSEC	9152
@@RDSSC	941C @@READ	92B7 @@REMOV	91E5
@@RENAM	91D0 @@REW	92CC @@RMTSK	9056
@@RPTSK	906B @@RREAD	92E1 @@RSLCT	913D
@@RSTOR	90FE @@RUN	9389 @@RWRIT	92F6
@@SEEK	9128 @@SEEKSC	930B @@SKIP	9320
@@SLCT	90E9 @@STEPI	9113 @@TIME	8F84
@@VDCTL	8F30 @@VER	9335 @@VRSEC	9167
@@WEOF	934A @@WHERE	8E25 @@WRITE	935F
@@WRSEC	9191 @@WRSSC	91A6 @@WTRK	91BB

2400 is the transfer address

00000 Total errors

NOTES:

NOTES:

KSM/FLT - Keystroke multiply filter

The KSM filter allows multiple characters or lines to be assigned to an alphabetic key. It must be installed with the SET and FILTER Library commands. It will install in high memory, and will not attempt to use the low driver zone.

```

00100 ;KSM/ASM - Keystroke Multiply Filter
0000 00110 TITLE <KSM/FLT - LS-DOS 6.2>
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
00150 ;
0000 00160 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVCMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
0000 00000 *LIST ON
00170 *GET COPYCOM:3 ;Copyright message
0000 00320 ; COPYCOM - File for Copyright COMMENT block
00330 ;
0000 00340 COM '<*(C) 1982,83,84 by LSI*>'
00180 ;
2400 00190 ORG 2400H
00200 ;
00210 KSM
2400 00220 @@CKBRKC ;Check for break
2400 3E6A 00001 LD A,106
2402 EF 00002 RST 40
2403 2804 00230 JR Z,KSMA ;Contine if not
2405 21FFFF 00240 LD HL,-1 ; else abort
2408 C9 00250 RET
00260 ;
2409 ED53CA27 00270 KSMA LD (KSMDCB),DE ;Save ptr to DCB
240D E5 00280 PUSH HL ;Save ptr to cmdline buf
240E 00290 @@DSPLY HELLO$ ;Display copyright msg
00003 IFEQ 01H,1
240E 217225 00004 LD HL,HELLO$
00005 ENDIF
2411 3E0A 00006 LD A,10
2413 EF 00007 RST 40
2414 00300 @@FLAGS ;Get flags
2414 3E65 00008 LD A,101
2416 EF 00009 RST 40
2417 E1 00310 POP HL ;Rcvr cmdline pointer
00320 ;
00330 ; Check if entry from SET command
00340 ;
2418 FDCB025E 00350 BIT 3,(IY+'C'-'A') ;System request?
241C CA4B25 00360 JP Z,VIASET ;Quit if not
241F 11A126 00370 LD DE,KSMFCB ;Point to FCB
2422 00380 @@FSPEC ;Fetch the KSM filespec
2422 3E4E 00010 LD A,78
2424 EF 00011 RST 40
2425 C25725 00390 JP NZ,SPCREQ ;Jump on bad spec
2428 D5 00400 PUSH DE ;Save FCB pointer
2429 119626 00410 LD DE,PRMTBL$ ;Load param table pointer
242C 00420 @@PARAM ;Parse parms
242C 3E11 00012 LD A,17
242E EF 00013 RST 40
242F D1 00430 POP DE ;Recover FCB pointer
2430 C26325 00440 JP NZ,IOERR ;Go on parm error
2433 217225 00450 LD HL,DFTKSM ;Init to default ext
2436 00460 @@FEXT ;Fetch if not entered
2436 3E4F 00014 LD A,79
2438 EF 00015 RST 40
00470 ;
00480 ; Transfer requested ENTER char to test loc
00490 ;

```

```

2439 213B00 00500 EPARM LD HL,';' ; set default ";"
243C 3A9D26 00510 LD A,(ERSP) ;Test parm response
243F CB77 00520 BIT 6,A ;Flag is no good!
2441 C26125 00530 JP NZ,PRMERR
2444 CB6F 00540 BIT 5,A ;Test string or value
2446 7E 00550 LD A,(HL) ;P/u assumed string
2447 2001 00560 JR NZ,$+3 ;Go if string entry
2449 7D 00570 LD A,L ;P/u hex or dec entry
244A 321328 00580 LD (ECHAR+1),A ;Stuff it in there
244D D5 00590 PUSH DE
244E 116D25 00600 LD DE,KSM$ ;Check if filter is
2451 00610 @@GTMOD ; already resident
2451 3E53 00016 LD A,83
2453 EF 00017 RST 40
2454 227D24 00620 LD (KSMMEM+1),HL ;Stuff start
2457 EB 00630 EX DE,HL ;Put DCB ptr to HL
2458 D1 00640 POP DE
2459 202C 00650 JR NZ,OPENKSM ;Go if not
00660 ;
00670 ;
00680 ;
                Make sure that the new DCB is same as the old
245B E5 00690 PUSH HL ;Save where to stuff
245C 4E 00700 LD C,(HL) ;P/u DCB pointer LSB
245D 23 00710 INC HL
245E 46 00720 LD B,(HL) ;P/u DCB pointer MSB
245F 210600 00730 LD HL,6 ;Get old DCB name &
2462 09 00740 ADD HL,BC ; stuff into error
2463 7E 00750 LD A,(HL) ; message in case
2464 2C 00760 INC L ; a different DCB
2465 66 00770 LD H,(HL) ; is referenced
2466 6F 00780 LD L,A
2467 227226 00790 LD (DCBNAM$),HL
246A B4 00800 OR H ;If DCB name is null,
246B 2ACA27 00810 LD HL,(KSMDCB)
246E E5 00820 PUSH HL ;Save pointer to stuff
246F 2803 00830 JR Z,UPDPTR ; then OK to use
2471 B7 00840 OR A
2472 ED42 00850 SBC HL,BC ;Same DCB pointer?
2474 C1 00860 UPDPTR POP BC ;Rcvr pointer to stuff
2475 E1 00870 POP HL ;Rcvr address to put pointer
2476 C24F25 00880 JP NZ,DCBERR ;Quit if filter in use
00890 ;
00900 ;
00910 ;
                Same DCB - Okay to stuff
2479 71 00920 LD (HL),C ;Store the DCB pointer
247A 23 00930 INC HL
247B 70 00940 LD (HL),B
247C 210000 00950 KSMMEM LD HL,$-$ ;If res, ptr to start
247F 015200 00960 LD BC,ECHAR-DVRBGN+1
2482 09 00970 ADD HL,BC ;Resident, stuff ECHAR
2483 3A1328 00980 LD A,(ECHAR+1) ; where it is in memory
2486 77 00990 LD (HL),A ;Stuff in upper mem
2487 21C126 01000 OPENKSM LD HL,KSMBUF ;Pt to buffer area
248A 0600 01010 LD B,0 ;Init LRL=256
248C 01020 @@OPEN ;Open the file
248C 3E3B 00018 LD A,59
248E EF 00019 RST 40
248F C26325 01030 JP NZ,IOERR ;Jump on open error
2492 212128 01040 LD HL,DVREND ;Place file in memory 1st
2495 061A 01050 LD B,26 ;Init for 26 lines
2497 01060 KSM1 @@GET ;Get a char from file

```

```

2497 3E03      00020      LD      A,3
2499 EF        00021      RST      40
249A 200C      01070      JR       NZ,KSM2      ;Jump on error
249C 77        01080      LD       (HL),A      ;Stuff into memory
249D 23        01090      INC      HL      ;Inc memory pointer
249E FE0D      01100      CP       CR      ;Found end-of-line?
24A0 20F5      01110      JR       NZ,KSM1      ;Loop if not
24A2 10F3      01120      DJNZ     KSM1      ;Decrement the A-Z loop
24A4 2B        01130      DEC      HL      ;Backup over last CR &
24A5 04        01140      INC      B      ; adjust for one more
24A6 3E1C      01150      LD       A,1CH      ;No error here, just EOF
24A8 F5        01160      KSM2     PUSH    AF      ;Save error code
24A9           01170      @CLOSE    ;Close the file
24A9 3E3C      00022      LD       A,60
24AB EF        00023      RST      40
24AC F1        01180      POP      AF
24AD FE1C      01190      CP       1CH      ;Ck for eof
24AF C26325    01200      JP       NZ,IOERR      ;Jump on not eof error
24B2 360D      01210      KSM3     LD       (HL),CR      ;End with a <ENTER>
24B4 23        01220      INC      HL      ;For all remaining
24B5 10FB      01230      DJNZ     KSM3      ;"letters" not entered
24B7 DD2ACA27  01240      LD       IX,(KSMDCB)      ;Rcvr user DCB entry
24BB 112128    01250      LD       DE,DVREND      ;Calculate the length
24BE AF        01260      XOR      A      ; of the KSM file just
24BF ED52      01270      SBC      HL,DE      ; loaded
24C1 44        01280      LD       B,H      ;Xfer length
24C2 4D        01290      LD       C,L
24C3 2A7D24    01300      LD       HL,(KSMMEM+1)      ;If not previously res,
24C6 7D        01310      LD       A,L      ; move to HIGH$
24C7 B4        01320      OR       H
24C8 281E      01330      JR       Z,MOVTOHI
24CA C5        01340      PUSH     BC      ;Save length
24CB E5        01350      PUSH     HL      ;Save old start
24CC 09        01360      ADD      HL,BC      ;Start + data
24CD 3812      01370      JR       C,KSM3A      ;Bad if wrap past 0
24CF 016100    01380      LD       BC,DVREND-DVRBGN+1
24D2 09        01390      ADD      HL,BC      ;Start + data + filter
24D3 380C      01400      JR       C,KSM3A      ;Bad if wrap past 0
24D5 EB        01410      EX       DE,HL      ;Save in reg DE
24D6 E1        01420      POP      HL      ;Rcvr old start
24D7 23        01430      INC      HL      ;Pt to last byte used
24D8 23        01440      INC      HL
24D9 7E        01450      LD       A,(HL)      ;P/u last byte used
24DA 23        01460      INC      HL
24DB 66        01470      LD       H,(HL)      ; into HL
24DC 6F        01480      LD       L,A
24DD E5        01490      PUSH     HL
24DE AF        01500      XOR      A      ;Clear carry flag
24DF ED52      01510      SBC      HL,DE      ;Is req > available?
24E1 E1        01520      KSM3A    POP      HL      ;Rcvr old start to reuse
24E2 C1        01530      POP      BC      ;Rcvr length of req
24E3 DA5325    01540      JP       C,NOROOM      ;Quit if file too big
24E6 1809      01550      JR       KSM0A
24E8 C5        01560      MOVTOHI  PUSH    BC      ;Save data length
24E9 210000    01570      LD       HL,0      ;P/u current high memory
24EC 45        01580      LD       B,L
24ED           01590      @HIGH$
24ED 3E64      00024      LD       A,100
24EF EF        00025      RST      40
24F0 C1        01600      POP      BC      ;Recover data length
24F1 22C327    01610      KSM0A    LD       (DVRBGN+2),HL      ;Stuff last byte used

```

The Source	UTILITY Files	KSM/FLT - LS-DOS 6.2	Page 00004
24F4 22CF27	01620	LD (RX1),HL ;Stuff ptr to flag byte	
24F7 3600	01630	LD (HL),0 ;Init the KSM char ptr	
24F9 2B	01640	DEC HL ; to zero to show no	
24FA 3600	01650	LD (HL),0 ; char avail at startup	
24FC 2B	01660	DEC HL	
24FD 112128	01670	LD DE,DVREND ;Move data to high	
2500 1A	01680	LD A,(DE) ;Data is in reverse order	
2501 77	01690	LD (HL),A	
2502 2B	01700	DEC HL ;Dec himem ptr	
2503 13	01710	INC DE ; and inc the char ptr	
2504 0B	01720	DEC BC ;Reduce char count	
2505 78	01730	LD A,B ; and check if done	
2506 B1	01740	OR C	
2507 20F7	01750	JR NZ,MOVLP ;Loop back if not	
2509 016000	01760	LD BC,DVREND-DVRBGN ;Get driver len	
250C AF	01770	XOR A ;Reduce potential HIGH\$	
250D ED42	01780	SBC HL,BC ; by driver length	
250F 3A7D24	01790	LD A,(KSMEM+1) ;Don't update HIGH\$	
2512 B7	01800	OR A ; if previously res	
2513 2809	01810	JR Z,DOHIGH ;Go if not resident	
	01820 ;		
	01830 ;		
	01840 ;	Module already resident	
2515 ED5B7D24	01850	LD DE,(KSMEM+1) ;P/u module entry point	
2519 213926	01860	LD HL,KSMRPL\$; & reuse the filter	
251C 1818	01870	JR KSM8	
	01880 ;		
	01890 ;		
	01900 ;	Stuff new HIGH\$ value (Note: B=0 for driver	
	01910 ;	length so there is no damage on the @@HIGH\$ SVC	
251E 0600	01920	LD B,0	
2520	01930	@@HIGH\$	
2520 3E64	00026	LD A,100	
2522 EF	00027	RST 40	
2523 23	01940	INC HL ;Pt to driver start	
2524 EB	01950	EX DE,HL	
2525 D5	01960	PUSH DE ;Save start of driver	
2526 210900	01970	LD HL,KSMDCB-DVRBGN	
2529 19	01980	ADD HL,DE ;Point to filter DCB ptr	
252A 22DF27	01990	LD (RX2),HL	
252D 21C127	02000	LD HL,DVRBGN ;Move parms also	
2530 EDB0	02010	LDIR	
2532 D1	02020	POP DE ;Rcvr driver ept	
2533 212226	02030	LD HL,KSMACT\$;Init "KSM installed	
2536 DD360045	02040	LD (IX),40H15 ;Init DCB type to "input"	
253A DD7301	02050	LD (IX+1),E ; & filter & stuff the	
253D DD7202	02060	LD (IX+2),D ; filter address	
2540 FDCB03F6	02070	SET 6,(IX+'D'-'A') ;Turn on device flag bit	
2544	02080	@@LOGOT ;Display installation msg	
	00028	IFEQ 00H,1	
	00029	LD HL,	
	00030	ENDIF	
2544 3E0C	00031	LD A,12	
2546 EF	00032	RST 40	
2547 210000	02090	LD HL,0 ;Set no error	
254A C9	02100	RET ;Back to the user	
	02110 ;		
	02120 ;	Error processing	
	02130 ;		
254B 21FB25	02140	LD HL,VIASET\$;"Install with Set	
254E DD	02150	DB 0DDH	


```

The Source      UTILITY Files      KSM/FLT - LS-DOS 6.2      Page 00005

254F 215226    02160 DCBERR LD      HL,DCBERR$      ;"Filter in use already
2552 DD        02170          DB      0DDH
2553 217526    02180 NOROOM LD     HL,NOROOM$      ;"Memory frozen
2556 DD        02190          DB      0DDH
2557 211026    02200 SPCREQ LD     HL,SPCREQ$      ;"Missing filespec
255A           02210          @LOGOT      ;Display an error
           00033          IFEQ      00H,1
           00034          LD        HL,
           00035          ENDF
255A 3E0C      00036          LD        A,12
255C EF        00037          RST      40
255D 21FFFF    02220          LD        HL,-1      ;Set abort code
2560 C9        02230          RET
2561 3E2C      02240 PRMERR LD     A,44      ;Init PARM ERROR
2563 6F        02250 IOERR  LD     L,A      ;Error code to HL
2564 2600      02260          LD        H,0
2566 F6C0      02270          OR        0C0H      ;Set short, return
2568 4F        02280          LD        C,A      ;Error to C
2569           02290          @ERROR      ; for error display
2569 3E1A      00038          LD        A,26
256B EF        00039          RST      40
256C C9        02300          RET
           02310 ;
           02320 ;      Data and message area
           02330 ;
256D 24        02340 KSM$   DB      '$KSM',3
      4B 53 4D 03
2572           02350 DFTKSM EQU      $      ;Note: HELLO$ must follow
2572 4B        02360 HELLO$ DB      'KSM Filter'
      53 4D 20 46 69 6C 74 65
      72
257C           02370 *GET     CLIENT:3
           03950 ;CLIENTS/ASM - File to establish sign-on headers
           03960 ;
257C 20        03970          DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
      2D 20 36 2E 32 2E 30 20
      2D 20 43 6F 70 79 72 69
      67 68 74 20 31 39 38 32
      2F 38 33 2F 38 34 20 62
      79 20 4C 6F 67 69 63 61
      6C
25A6 20        03980          DB      ' Systems, Inc.      ',10
      53 79 73 74 65 6D 73 2C
      20 49 6E 63 2E 20 20 20
      20 20 20 0A
           03990 ;
25BB 41        04000          DB      'All Rights Reserved. Licensed 1982/83/84'
      6C 6C 20 52 69 67 68 74
      73 20 52 65 73 65 72 76
      65 64 2E 20 4C 69 63 65
      6E 73 65 64 20 31 39 38
      32 2F 38 33 2F 38 34
25E3 20        04010          DB      ' to xxxxxxxxxxxxxxxxxxx',10,13
      74 6F 20 78 78 78 78 78
      78 78 78 78 78 78 78 78
      78 78 78 78 78 0A 0D
           02380 ;
25FB 4D        02390 VIASET$ DB      'Must install via SET',CR
      75 73 74 20 69 6E 73 74
      61 6C 6C 20 76 69 61 20
      53 45 54 0D

```

The Source	UTILITY Files	KSM/FLT - LS-DOS 6.2	Page 00006
2610 46	02400 SPCREQ\$ DB	'Filespec required',CR	
69 6C 65	73 70 65 63 20		
72 65 71	75 69 72 65 64		
0D			
2622 4B	02410 KSMACT\$ DB	'KSM is now operational',CR	
53 4D 20	69 73 20 6E 6F		
77 20 6F	70 65 72 61 74		
69 6F 6E	61 6C 0D		
2639 4B	02420 KSMRPL\$ DB	'KSM filter data replaced',CR	
53 4D 20	66 69 6C 74 65		
72 20 64	61 74 61 20 72		
65 70 6C	61 63 65 64 0D		
2652 4B	02430 DCBERR\$ DB	'KSM filter already attached to *xx',CR	
53 4D 20	66 69 6C 74 65		
72 20 61	6C 72 65 61 64		
79 20 61	74 74 61 63 68		
65 64 20	74 6F 20 2A 78		
78 0D			
2672	02440 DCBNAM\$ EQU	\$-3	
2675 52	02450 NOROOM\$ DB	'Request exceeds available memory',CR	
65 71 75	65 73 74 20 65		
78 63 65	65 64 73 20 61		
76 61 69	6C 61 62 6C 65		
20 6D 65	6D 6F 72 79 0D		
2696 D2	02460 PRMTBL\$ DB	'R'!80H,0F5H,'ENTER',0	
F5 45 4E	54 45 52 00		
269D	02470 ERSP EQU	\$-1	
	02480 ;		
269E 3A24	02490 DW	EPARM+1	
26A0 00	02500 DB	0	
	02510 ;		
0020	02520 KSMFCB DEFS	32	
0100	02530 KSMBUF DEFS	256	
	02540 ;		
	02550 ;		
	02560 ;		
		Key-Stroke Multiplication driver	
27C1 180B	02570 DVRBGN JR	START	;Branch around header
27C3 0000	02580 DW	\$-\$;Last byte used
27C5 04	02590 DB	4,'\$KSM'	
24 4B 53 4D			
27CA 0000	02600 KSMDCB DW	\$-\$;Pointer to KSM's DCB
27CC 0000	02610 DW	0	
	02620 ;		
27CE 210000	02630 START LD	HL,0	;P/u possible address to
27CF	02640 RX1 EQU	\$-2	
27D1 56	02650 LD	D,(HL)	; a KSM that was parsed
27D2 2B	02660 DEC	HL	; to a ';' logical ENTER
27D3 5E	02670 LD	E,(HL)	;If this vector is zero,
27D4 2B	02680 DEC	HL	; no KSM continuation is
27D5 EB	02690 EX	DE,HL	; pending - find a new
27D6 F5	02700 PUSH	AF	; entry. Save flags.
27D7 7C	02710 LD	A,H	; If <> 0, grab the KSM
27D8 B5	02720 OR	L	; line continuation
27D9 202B	02730 JR	NZ,DVR4A	
27DB F1	02740 POP	AF	;Rcvr flags
27DC D5	02750 PUSH	DE	;Save ptr to 'A'-KSM
27DD DD2ACA27	02760 DVR1 LD	IX,(KSMDCB)	;Chain to next DCB module
27DF	02770 RX2 EQU	\$-2	
27E1	02780 @@CHNIO		
27E1 3E14	00040 LD	A,20	
27E3 EF	00041 RST	40	

```

27E4 D1      02790      POP      DE      ;Rcvr 'A'-KSM pointer
27E5 C0      02800      RET      NZ      ;Back if nothing or error
27E6 CB7F    02810      BIT      7,A     ;Is it a CLEAR function?
27E8 C8      02820      RET      Z      ;Ret if <CLEAR> not down
27E9 F5      02830      PUSH     AF      ;Save key entry
27EA FEC1    02840      CP       'A'+80H ;Ck for range A-Z
27EC 3804    02850      JR       C,DVR2  ;Exit if < 'A'
27EE FEDB    02860      CP       'Z'+1+80H
27F0 3803    02870      JR       C,DVR3  ;Use it if A-Z
27F2 F1      02880      POP      AF      ;Rcvr orig flag
27F3 BF      02890      CP       A      ;Set Z-flag
27F4 C9      02900      RET
                02910 ;
                02920 ;      Key code entry includes <CLEAR> key
                02930 ;
27F5 F1      02940      POP      AF      ;Rcvr orig flag
27F6 62      02950      LD       H,D     ;Rcvr ptr to 'A'-KSM
27F7 6B      02960      LD       L,E     ; & xfer to reg HL
27F8 D6C1    02970      SUB      80H+'A' ;Adjust offset to index
27FA 280B    02980      JR       Z,DVR5  ;Bypass if was 'A'
27FC 47      02990      LD       B,A     ;Set loop counter
27FD 3E0D    03000      LD       A,CR    ;Read past the KSM lines
27FF BE      03010      CP       (HL)   ; for letters preceding
2800 2B      03020      DEC      HL     ; key entry to find the
2801 20FC    03030      JR       NZ,DVR4 ; KSM line for entered
2803 10FA    03040      DJNZ     DVR4   ; key code
2805 3E      03050      DB       3EH    ;Ignore next inst
                03060 ;
                03070 ;      Routine to pick up the next KSM character
                03080 ;      & return it to the system KI request
                03090 ;
2806 F1      03100      POP      AF      ;Clean the stack
2807 7E      03110      LD       A,(HL)  ;P/u the next KSM char
2808 2B      03120      DEC      HL     ;Dec pointer to next one
2809 EB      03130      EX       DE,HL  ;Put either a pointer to
280A 23      03140      INC      HL     ; the next KSM char or
280B FE0D    03150      CP       CR     ; if got last, zero the
280D 280B    03160      JR       Z,DVR6  ; data pointer
280F 73      03170      LD       (HL),E ;Stuff pointer to next
2810 23      03180      INC      HL     ; character to fetch
2811 72      03190      LD       (HL),D
2812 FE3B    03200      ECHAR   CP       ';'   ;Ck on logical line end
2814 2002    03210      JR       NZ,DVR7 ; & convert to <ENTER>
2816 3E0D    03220      LD       A,CR    ; if it was semi-colon
2818 BF      03230      CP       A      ;Tell the system we have
2819 C9      03240      RET      ; retrieved a char
                03250 ;
                03260 ;      Got the terminating X'0D' - Clear the pointer
                03270 ;
281A AF      03280      XOR      A      ;Clear the KSM char ptr
281B 77      03290      LD       (HL),A ; as next request is new
281C 23      03300      INC      HL
281D 77      03310      LD       (HL),A
281E FEFF    03320      CP       0FFH   ;Set NZ & A = 0
2820 C9      03330      RET
2821         03340      DVREND  EQU      $
                03350 ;
2400         03360      END      KSM

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @MOD2	0000 @MOD4	FFFF
CR	000D DCBERR	254F DCBERR\$	2652
DCBNAM\$	2672 DFTKSM	2572 DOHIGH	251E
DVR1	27DD DVR2	27F2 DVR3	27F5
DVR4	27FF DVR4A	2806 DVR5	2807
DVR6	281A DVR7	2818 DVRBGN	27C1
DVREND	2821 ECHAR	2812 EPARM	2439
ERSP	269D HELLO\$	2572 IOERR	2563
KSM	2400 KSM\$	256D KSM0A	24F1
KSM1	2497 KSM2	24A8 KSM3	24B2
KSM3A	24E1 KSM8	2536 KSMA	2409
KSMACT\$	2622 KSMBUF	26C1 KSMDCB	27CA
KSMFCB	26A1 KSMMEM	247C KSMRPL\$	2639
LF	000A MOVLP	2500 MOVTOHI	24E8
NOROOM	2553 NOROOM\$	2675 OPENKSM	2487
PRMERR	2561 PRMTBL\$	2696 RX1	27CF
RX2	27DF SPCREQ	2557 SPCREQ\$	2610
START	27CE UPDPTR	2474 VIASET	254B
VIASET\$	25FB @@ABORT	8E86 @@ADTSK	8F19
@@BANK	9431 @@BKSP	9111 @@BREAK	9447
@@CHNIO	8E71 @@CKBRKC	9495 @@CKDRV	8F6D
@@CKEOF	9126 @@CKTSK	8F04 @@CLOSE	90FC
@@CLS	947F @@CMNDI	8EB0 @@CMNDR	8EC5
@@CTL	8CD5 @@DATE	8E47 @@DCSTAT	8FAC
@@DEBUG	8EEF @@DECHEX	93B1 @@DIRRD	931E
@@DIRWR	9333 @@DIV16	939C @@DIV8	9387
@@DODIR	8F82 @@DSP	8C99 @@DSPLY	8D39
@@ERROR	8EDA @@EXIT	8E9B @@FEXT	928B
@@FLAGS	941B @@FNAME	92A0 @@FSPEC	9276
@@GATRD	9309 @@GATWR	9348 @@GET	8CAD
@@GTDCB	92CA @@GTDC	92B5 @@GTMOD	92DF
@@HDFMT	9054 @@HEX16	93F0 @@HEX8	93DB
@@HEXDEC	93C6 @@HIGH\$	9405 @@INIT	90D2
@@KBD	8D11 @@KEY	8C85 @@KEYIN	8D25
@@KLTSK	8F58 @@LOAD	924C @@LOC	913B
@@LOF	9150 @@LOGGER	8D70 @@LOGOT	8D85
@@MSG	8DBC @@MUL16	9372 @@MUL8	935D
@@OPEN	90E7 @@PARAM	8E32 @@PAUSE	8E1D
@@PEOF	9165 @@POSN	917A @@PRINT	8DD1
@@PRT	8CE9 @@PUT	8CC1 @@RAMDIR	8F97
@@RDSEC	902A @@RDSSC	92F4 @@READ	918F
@@REMOV	90BD @@RENAM	90A8 @@REW	91A4
@@RMTSK	8F2E @@RPTSK	8F43 @@RREAD	91B9
@@RSLCT	9015 @@RSTOR	8FD6 @@RUN	9261
@@RWRTIT	91CE @@SEEK	9000 @@SEEKSC	91E3
@@SKIP	91F8 @@SLCT	8FC1 @@STEPI	8FEB
@@TIME	8E5C @@VDCTL	8E08 @@VER	920D
@@VRSEC	903F @@WEOF	9222 @@WHERE	8CFD
@@WRITE	9237 @@WRSEC	9069 @@WRSSC	907E
@@WRTK	9093		

2400 is the transfer address

00000 Total errors

NOTES:

LOG/CMD - Log in a disk

The main use of the Log utility is to allow swapping a double sided disk into drive 0 after booting on a single sided disk.

```

00100 ;LOG/ASM - Optional Disk Log Program
00110 TITLE <LOG - LS-DOS 6.2>
00120 ;
00130 CR EQU 13
00140 LF EQU 10
00150 CRSON EQU 14
00160 ;
00170 *GET SVCMAC:3 ;SVC Macro equivalents
00180 ;SVC MAC/ASM - LS-DOS Version VI
00190 *LIST OFF
00200 *LIST ON
00210 *GET COPYCOM:3 ;Copyright message
00220 ; COPYCOM - File for Copyright COMMENT block
00230 ;
00240 COM '<*(C) 1982,83,84 by LSI*>'
00250 ;
00260 ORG 2600H
00270 ;
00280 LOG
00290 @CKBRKC ;Check for break
00300 LD A,106
00310 RST 40
00320 JR Z,LOGA ;Go if not
00330 LD HL,-1 ; else abort
00340 RET
00350 ;
00360 LOGA
00370 LD (STACK),SP ;Save entry SP
00380 PUSH HL ;Save cmdline ptr
00390 @DSPLY HELLO$ ;Display the signon msg
00400 IFEQ 01H,1
00410 LD HL,HELLO$
00420 ENDIF
00430 LD A,10
00440 RST 40
00450 POP HL ;Recover cmdline ptr
00460 ;
00470 ; Start of main module code
00480 ;
00490 START LD C,0 ;Default drive 0
00500 SKIPSP LD A,(HL) ;Scan command line
00510 INC HL
00520 CP ' ' ;Skip spaces
00530 JR Z,SKIPSP
00540 CP ':' ;Look for colon
00550 JR NZ,DEFAULT ;End of line if not found
00560 LD A,(HL) ;Get drive #
00570 SUB 30H ;Make a number
00580 JP C,ILLDRV ;# too low
00590 CP 7+1
00600 JP NC,ILLDRV ;# too hi
00610 LD C,A ;Save in C
00620 LD A,C ;Drive 0?
00630 AND A
00640 JR NZ,NOWAIT ;Go if not
00650 @DSPLY WAIT$ ;Display "Switch disks"
00660 IFEQ 01H,1
00670 LD HL,WAIT$
00680 ENDIF
00690 LD A,10
00700 RST 40

```

```

2637 2052      00530      JR      NZ,IOERR
2639           00540      @@KEY           ;Wait for a key
2639 3E01      00013      LD      A,1
263B EF       00014      RST      40
263C 204D      00550      JR      NZ,IOERR
263E C5       00560      PUSH     BC           ;Save the drive #
263F 0E0D      00570      LD      C,CR           ;Output a new line
2641           00580      @@DSP
2641 3E02      00015      LD      A,2
2643 EF       00016      RST      40
2644 C1       00590      POP      BC           ;Recover drive #
2645 2044      00600      JR      NZ,IOERR
2647 1807      00610      JR      NOCHK           ;Can't call CKDRV if :0
                00620 ;
2649           00630 NOWAIT @@CKDRV           ;Drive ready?
2649 3E21      00017      LD      A,33
264B EF       00018      RST      40
264C 3E20      00640      LD      A,32           ;"Illegal drive number"
264E 203B      00650      JR      NZ,IOERR           ;Go if not ready
2650 210028    00660 NOCHK LD      HL,BUFFER           ;Sector buffer
2653 110000    00670      LD      DE,0           ;Read boot sector
2656           00680      @@RDSEC
2656 3E31      00019      LD      A,49
2658 EF       00020      RST      40
2659 2030      00690      JR      NZ,IOERR           ;Go if error
265B           00700      @@GTDCT           ;Point IY to DCT
265B 3E51      00021      LD      A,81
265D EF       00022      RST      40
265E 23       00710      INC     HL           ;Point HL to byte 2
265F 23       00720      INC     HL
2660 7E       00730      LD      A,(HL)           ;Get dir cyl #
2661 FD7709    00740      LD      (IY+9),A           ; and put in DCT
                00750 ;
2664 57       00760      LD      D,A           ;Now read GAT
2665 210028    00770      LD      HL,BUFFER           ;Disk sector buffer
2668 5D       00780      LD      E,L           ;Set to 0
2669           00790      @@RDSEC
2669 3E31      00023      LD      A,49
266B EF       00024      RST      40
266C FE06      00800      CP      6           ;Must be sys sector
266E 201B      00810      JR      NZ,IOERR           ;Go if error
                00820 ;
2670 2ECD      00830      LD      L,0CDH           ;Offset to disk type
2672 7E       00840      LD      A,(HL)           ;P/U disk type
2673 E620      00850      AND     20H           ;Check # of sides bit
2675 47       00860      LD      B,A           ;Save in B
2676 FD7E04    00870      LD      A,(IY+4)           ;P/U byte in DCT
2679 E6DF      00880      AND     0DFH           ;Mask out old value
267B B0       00890      OR      B           ;Put in new value
267C FD7704    00900      LD      (IY+4),A           ;Put back in DCT
                00910 ;
267F 210000    00920      LD      HL,0           ;Set no error
2682 310000    00930 $QUIT LD      SP,$-$           ;P/u original stack
2683           00940 STACK EQU     $-2
2685           00950      @@CKBRKC           ;Clear any break
2685 3E6A      00025      LD      A,106
2687 EF       00026      RST      40
2688 C9       00960      RET           ;Back to the user
                00970 ;
2689 3E20      00980 ILLDRV LD      A,32           ;Init "illegal drv"
268B 6F       00990 IOERR LD      L,A           ;Put error # into HL

```



```

268C 2600      01000      LD      H,0
268E F6C0      01010      OR      0C0H      ;Abbrev, return
2690 4F        01020      LD      C,A        ;Error code to C
2691          01030      @@ERROR      ; for error display
2691 3E1A      00027      LD      A,26
2693 EF        00028      RST      40
2694 18EC      01040      JR      $QUIT
                01050      ;
2696 4C        01060      HELLO$ DB      'LOG Drive'
                4F 47 20 44 72 69 76 65
269F          01070      *GET      CLIENT:3
                03950      ;CLIENTS/ASM - File to establish sign-on headers
                03960      ;
269F 20        03970      DB      ' - 6.2.0 - Copyright 1982/83/84 by Logical'
                2D 20 36 2E 32 2E 30 20
                2D 20 43 6F 70 79 72 69
                67 68 74 20 31 39 38 32
                2F 38 33 2F 38 34 20 62
                79 20 4C 6F 67 69 63 61
                6C
26C9 20        03980      DB      ' Systems, Inc.      ',10
                53 79 73 74 65 6D 73 2C
                20 49 6E 63 2E 20 20 20
                20 20 20 0A
                03990      ;
26DE 41        04000      DB      'All Rights Reserved. Licensed 1982/83/84'
                6C 6C 20 52 69 67 68 74
                73 20 52 65 73 65 72 76
                65 64 2E 20 4C 69 63 65
                6E 73 65 64 20 31 39 38
                32 2F 38 33 2F 38 34
2706 20        04010      DB      ' to xxxxxxxxxxxxxxxxxxx',10,13
                74 6F 20 78 78 78 78 78
                78 78 78 78 78 78 78 78
                78 78 78 78 78 0A 0D
271E 45        01080      WAIT$ DB      'Exchange disks and depress <ENTER> ',3
                78 63 68 61 6E 67 65 20
                64 69 73 6B 73 20 61 6E
                64 20 64 65 70 72 65 73
                73 20 3C 45 4E 54 45 52
                3E 20 03
2800          01090      ORG      $<-8+1<8
2800          01100      BUFFER EQU      $
                01110      ;
2600          01120      END      LOG

```

\$QUIT	2682 @@1	0000 @@2	0000
@@3	0000 @@4	0000 @MOD2	0000
@MOD4	FFFF BUFFER	2800 CR	000D
CRSON	000E DEFAULT	262D HELLO\$	2696
ILLDRV	2689 IOERR	268B LF	000A
LOG	2600 LOGA	2609 NOCHK	2650
NOWAIT	2649 SKIPSP	2617 STACK	2683
START	2615 WAIT\$	271E @@ABORT	750F
@@ADTSK	75A2 @@BANK	7ABA @@BKSP	779A
@@BREAK	7AD0 @@CHNIO	74FA @@CKBRKC	7B1E
@@CKDRV	75F6 @@CKEOF	77AF @@CKTSK	758D
@@CLOSE	7785 @@CLS	7B08 @@CMNDI	7539
@@CMNDR	754E @@CTL	735E @@DATE	74D0
@@DCSTAT	7635 @@DEBUG	7578 @@DECHEX	7A3A
@@DIRRD	79A7 @@DIRWR	79BC @@DIV16	7A25
@@DIV8	7A10 @@DODIR	760B @@DSP	7322
@@DSPLY	73C2 @@ERROR	7563 @@EXIT	7524
@@FEXT	7914 @@FLAGS	7AA4 @@FNAME	7929
@@FSPEC	78FF @@GATRD	7992 @@GATWR	79D1
@@GET	7336 @@GTDCB	7953 @@GTDC	793E
@@GTMOD	7968 @@HDFMT	76DD @@HEX16	7A79
@@HEX8	7A64 @@HEXDEC	7A4F @@HIGH\$	7A8E
@@INIT	775B @@KBD	739A @@KEY	730E
@@KEYIN	73AE @@KLTSK	75E1 @@LOAD	78D5
@@LOC	77C4 @@LOF	77D9 @@LOGGER	73F9
@@LOGOT	740E @@MSG	7445 @@MUL16	79FB
@@MUL8	79E6 @@OPEN	7770 @@PARAM	74BB
@@PAUSE	74A6 @@PEOF	77EE @@POSN	7803
@@PRINT	745A @@PRT	7372 @@PUT	734A
@@RAMDIR	7620 @@RDSEC	76B3 @@RDSSC	797D
@@READ	7818 @@REMOV	7746 @@RENAM	7731
@@REW	782D @@RMTSK	75B7 @@RPTSK	75CC
@@RREAD	7842 @@RSLCT	769E @@RSTOR	765F
@@RUN	78EA @@RWRT	7857 @@SEEK	7689
@@SEEKSC	786C @@SKIP	7881 @@SLCT	764A
@@STPEI	7674 @@TIME	74E5 @@VDCTL	7491
@@VER	7896 @@VRSEC	76C8 @@WEOF	78AB
@@WHERE	7386 @@WRITE	78C0 @@WRSEC	76F2
@@WRSSC	7707 @@WRTRK	771C	

2600 is the transfer address
00000 Total errors

NOTES:

MEMDISK/DCT - Memory disk driver

The Memdisk DCT program will establish a psuedo disk drive either in the main memory or in the alternate memory banks, if available. There must be room for Memdisk in the low driver zone or the installation will abort.

```

00100 ;MEMDISK/ASM - Memory Disk Driver
0000 00110 TITLE <MEMDISK/DCT - LS-DOS 6.2>
00120 ;
0000 00130 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00140 *GET VALUES:3 ;Misc. equates
03920 ;VALUES/ASM - Version 6
03930 *LIST OFF
04200 *LIST ON
0000 00150 *GET COPYCOM:3 ;Copyright message
04210 ; COPYCOM - File for Copyright COMment block
04220 ;
0000 04230 COM '<*(C) 1982,83,84 by LSI*>'
00160 ;
0A00 00170 SDBPC EQU 5*2*256 ;Single Density Bytes/Cyl
1200 00180 DDBPC EQU 6*3*256 ;Double Density Bytes/Cyl
8000 00190 LOWEST EQU 8000H ;Lowest addr for Bank 0
1300 00200 HIDRVR EQU 1300H ;Highest addr for Driver
2300 00210 BUFFER$ EQU 2300H ;Temporary I/O buffer Add
0003 00220 MINCYL EQU 3
000F 00230 WP EQU 15 ;Write Prot Disk Error #
00240 ;
2C00 00250 ORG 2C00H
00260 ;
00270 START
2C00 00280 @@CKBRKC ;Check for break
2C00 3E6A 00001 LD A,106
2C02 EF 00002 RST 40
2C03 2804 00290 JR Z,STARTA ;Continue if not
2C05 21FFFF 00300 LD HL,-1 ; else abort
2C08 C9 00310 RET
00320 ;
2C09 00330 STARTA EQU $
2C09 ED73222C 00340 LD (EXIT+1),SP ;Save SP location
00350 ;
00360 ; Install or Disable MemDISK
00370 ;
2C0D CDF62C 00380 CALL CALCDRV ;Calculate drive #
2C10 CD3F30 00390 CALL DOMEM ;Get type of memdisk
2C13 CD9A2E 00400 CALL INSTMEM ;Install MemDISK
00410 ;
00420 ; Exit - Clean stack, Set HL, Revector <BREAK>
00430 ;
2C16 210000 00440 NORMEX LD HL,0 ;Normal Exit - HL = 0
2C19 1806 00450 JR EXIT ;Get SP & RETURN
00460 ;
2C1B CD2C2D 00470 ABORT CALL GETDUP ;Get duplicate DCT
2C1E 21FFFF 00480 LD HL,-1 ;Abort
00490 ;
2C21 310000 00500 EXIT LD SP,$-$ ;P/u SP address
2C24 00510 @@CKBRKC ;Clear break
2C24 3E6A 00003 LD A,106
2C26 EF 00004 RST 40
2C27 C9 00520 RET
00530 ;
2C28 00540 *GET MEMDISKB:3
04240 ;MEMDISKB/ASM - Miscellaneous Subroutines
2C28 04250 SUBTTL '<MEMDISKB - Subroutines>'

```

MEMDISKB - Subroutines

```

2C28      04260      PAGE
          04270 ;
          04280 ;      SETBANK - Tell system which banks are used
          04290 ;
2C28 3E00      04300 SETBANK LD      A,$-$      ;P/u bank #
2C2A 4F        04310      LD      C,A          ;Xfer to C
2C2B FE03      04320      CP      3            ;Both banks 1 & 2 ?
2C2D 2005      04330      JR      NZ,STBANK     ;No - just 1 bank
2C2F 0D        04340      DEC     C            ;Set C = 2
2C30 CD342C    04350      CALL    STBANK        ;Show Bank in use
2C33 0D        04360      DEC     C            ;C = 1
2C34 C5        04370 STBANK PUSH    BC          ;Save BC
2C35 0603      04380      LD      B,3          ;Show in use function #
2C37          04390      @BANK                  ;Let system know it
2C37 3E66      00005      LD      A,102
2C39 EF        00006      RST     40
2C3A C1        04400      POP     BC
2C3B C9        04410      RET                    ;RETurn
          04420 ;
          04430 ;      FREBANK - Free up Bank C
          04440 ;
2C3C C5        04450 FREBANK PUSH    BC          ;Save C & B
2C3D 0601      04460      LD      B,1          ;Show bank available
2C3F          04470      @BANK
2C3F 3E66      00007      LD      A,102
2C41 EF        00008      RST     40
2C42 C1        04480      POP     BC          ;Recover C
2C43 C9        04490      RET                    ;RETurn
          04500 ;
          04510 ;      DECASC2 - Display Number to video
          04520 ;
2C44 CDAE2C    04530 DECASC2 CALL    SAVEREG     ;Save Registers
2C47 F5        04540      PUSH    AF          ;Save #
2C48 0E08      04550      LD      C,BS        ;Backspace
2C4A CD592C    04560      CALL    DSP          ;Output byte
2C4D CD592C    04570      CALL    DSP          ;Twice
2C50 F1        04580      POP     AF          ;Recover A
2C51 CD632C    04590      CALL    DECASC      ;Convert to ASCII
2C54 4C        04600      LD      C,H        ;P/u ms digit
2C55 CD592C    04610      CALL    DSP
2C58 4D        04620      LD      C,L        ;P/u ls digit
          04630 ;
          04640 ;      DSP - Output byte to Video & exit if I/O err
          04650 ;
2C59          04660 DSP      @DSP                  ;Output byte
2C59 3E02      00009      LD      A,2
2C5B EF        00010      RST     40
2C5C C8        04670      RET     Z            ;RETurn if good
          04680 ;
          04690 ;      IOERR - Set HL = Error # & Abort
          04700 ;
2C5D 6F        04710 IOERR  LD      L,A        ;Set HL = I/O Error #
2C5E 2600      04720      LD      H,0
2C60 C3212C    04730      JP      EXIT        ;Go to exit routine
          04740 ;
          04750 ;      Display Decimal ASCII equivalent
          04760 ;
2C63 262F      04770 DECASC LD      H,2FH     ;H=msb of BCD ASCII
2C65 24        04780 LPADD  INC     H        ;Bump msb
2C66 D60A      04790      SUB     10        ;Successive sub's of 10

```

MEMDISKB - Subroutines

```

2C68 30FB      04800      JR      NC,LPADD      ;Keep sub til carry
2C6A C63A      04810      ADD      A,3AH      ;A = 1sb ASCII
2C6C 6F        04820      LD        L,A      ;HL => DEC ASCII
2C6D C9        04830      RET
                04840      ;
                04850      ;      DECHEX - Convert Decimal ASCII to Hex
                04860      ;
2C6E CD822C    04870      DECHEX  CALL     GETDIG      ;Get digit
2C71 23        04880      INC      HL      ;Next byte in buffer
2C72 05        04890      DEC      B      ;Dec digit counter
2C73 280B      04900      JR      Z,DONE1      ;All done
2C75 57        04910      LD        D,A      ;Xfer to D
2C76 CD822C    04920      CALL     GETDIG      ;Get digit
2C79 5F        04930      LD        E,A      ;Save digit
2C7A 7A        04940      LD        A,D      ;P/u ten's digit
2C7B 87        04950      ADD      A,A      ;Multiply
2C7C 87        04960      ADD      A,A      ; A times 10
2C7D 82        04970      ADD      A,D      ; and add it
2C7E 87        04980      ADD      A,A      ; to the ones digit
2C7F 83        04990      ADD      A,E      ;A = number of tracks
2C80 BF        05000      CP        A      ;Set Z flag
2C81 C9        05010      RET      ; and RETURN
                05020      ;
2C82 7E        05030      GETDIG  LD        A,(HL)      ;P/u second digit
2C83 D630      05040      SUB      '0'      ;Cvt to binary
2C85 3803      05050      JR      C,ILLEGAL      ;Clr stack & RETURN NZ
2C87 FE0A      05060      CP        10      ;Legal digit
2C89 D8        05070      RET      C      ;Yes - A = digit
2C8A 3C        05080      ILLEGAL INC      A      ;Reset Z flag
2C8B E1        05090      POP      HL      ;Clear stack
2C8C C9        05100      RET      ; and RETURN
                05110      ;
                05120      ;      Verify Error - P/u Bank / Address & display
                05130      ;
2C8D E5        05140      ERROR   PUSH     HL      ;L = 1sb of Address
2C8E 3EC9      05150      LD        A,0C9H      ;Modify GETADR routine
2C90 32722E    05160      LD        (STFRET),A      ;HL <= page from DE
2C93 CD5D2E    05170      CALL     GETADR
2C96 D1        05180      POP      DE      ;E = 1sb of address
2C97 6B        05190      LD        L,E      ;HL = Bad RAM address
                05200      ;
                05210      ;      Stuff Bank # and Address into string
                05220      ;
2C98 3E30      05230      LD        A,'0'      ;Cvt BANK # to ASCII
2C9A 81        05240      ADD      A,C      ;
2C9B 324737    05250      LD        (VBANK),A      ;Stuff into string
2C9E EB        05260      EX        DE,HL      ;Xfer address to DE
2C9F 215737    05270      LD        HL,VLOC      ;HL => string destination
2CA2           05280      @@HEX16      ;Cvt DE to Hex ASCII @ HL
2CA2 3E63      00011      LD        A,99
2CA4 EF        00012      RST      40
                05290      ;
                05300      ;      Display string & restore hi/low mem
                05310      ;
2CA5 213137    05320      LD        HL,BADRAM      ;"BAD RAM ..."
2CA8           05330      @LOGOT      ;Display it
                00013      IFEQ     00H,1
                00014      LD        HL,
                00015      ENDIF

```

MEMDISKB - Subroutines

```

2CA8 3E0C      00016      LD      A,12
2CAA EF        00017      RST      40
2CAB C3EE2E    05340      JP       OLDRV      ;Leave & clear stack
                05350 ;
                05360 ;      SAVEREG - Save All Primary Registers
                05370 ;
2CAE E3        05380      SAVEREG EX      (SP),HL
2CAF 22C42C    05390      LD      (RETADDR+1),HL
2CB2 E1        05400      POP      HL
2CB3 E5        05410      PUSH     HL
2CB4 ED53B832  05420      LD      (SAVEDE),DE
2CB8 D5        05430      PUSH     DE
2CB9 C5        05440      PUSH     BC
2CBA F5        05450      PUSH     AF
2CBB 11C62C    05460      LD      DE,RESTREG
2CBE D5        05470      PUSH     DE
2CBF ED5BB832  05480      LD      DE,(SAVEDE)
2CC3 C30000    05490      RETADDR JP      $-$
2CC6 F1        05500      RESTREG POP     AF
2CC7 C1        05510      POP      BC
2CC8 D1        05520      POP      DE
2CC9 E1        05530      POP      HL
2CCA C9        05540      RET
                05550 ;
                05560 ;      CKBANK - Check if Bank C is in use
                05570 ;
2CCB C5        05580      CKBANK PUSH     BC      ;Save BC
2CCC 0602      05590      LD      B,2      ;Bank in use ?
2CCE          05600      @BANK      ;Check it out
2CCE 3E66      00018      LD      A,102
2CD0 EF        00019      RST      40
2CD1 C1        05610      POP      BC      ;Recover BC
2CD2 C8        05620      RET      Z      ;RETurn if available
2CD3 C3DA32    05630      JP      BNKUSE    ; else - display "in use"
                05640 ;
                05650 ;      INPUT - Input a line to the input buffer
                05660 ;
2CD6 210039    05670      INPUT LD      HL,BUFFER    ;HL => Input buffer
2CD9          05680      @KEYIN    ;Input line
2CD9 3E09      00020      LD      A,9
2CDB EF        00021      RST      40
2CDC DA1B2C    05690      JP      C,ABORT    ;Exit if <BREAK> hit
2CDF 04        05700      INC      B      ;Set Z if no chars
2CE0 05        05710      DEC      B
2CE1 C9        05720      RET      ; else RETurn
                05730 ;
                05740 ;      GETCYL - Get max # of cylinders in A
                05750 ;
2CE2 D5        05760      GETCYL PUSH     DE      ;Save regs
2CE3 E5        05770      PUSH     HL
                05780 ;
                05790 ;      Init DE = # bytes/cyl, A = dividend (-1)
                05800 ;
2CE4 110012    05810      BPC      LD      DE,DBBPC    ;P/u bytes/cyl
2CE7 3EFF      05820      LD      A,-1    ;Init avail cyl cnt = -1
                05830 ;
                05840 ;      Divide total bytes available by Bytes/cyl
                05850 ;
2CE9 3C        05860      DIVLP  INC      A      ;Bump cyl count

```


MEMDISKB - Subroutines

```

2CEA B7      05870      OR      A
2CEB ED52    05880      SBC      HL,DE      ;Take off 1 cyl
2CED 30FA    05890      JR       NC,DIVLP    ;Loop until carry
                05900 ;
                05910 ;      A = # of cyls avail, Restore regs
                05920 ;
2CEF E1      05930      POP      HL          ;Recover regs
2CF0 D1      05940      POP      DE
                05950 ;
                05960 ;      Set Z flag if more than 1 cylinder available
                05970 ;
2CF1 FE02    05980      CP       2          ;0 or 1 ?
2CF3 D8      05990      RET      C          ;Yes - RETurn NZ
2CF4 BF      06000      CP       A          ;Set Z flag
2CF5 C9      06010      RET              ; and RETurn
                06020 ;
                06030 ;      CALCDRV - Calculate drive Number for MemDISK
                06040 ;
                06050 ;      DE => DCT block for Drive
                06060 ;
2CF6          06070      CALCDRV EQU      $
2CF6 EB      06080      EX       DE,HL      ;Xfer to HL
2CF7 22BA32  06090      LD       (SAVEDCT),HL ;Save DCT pointer
2CFA CD202D  06100      CALL    SAVDCT      ;Save DCT
2CFD 7C      06110      LD       A,H        ;Drive number issued ?
2CFE B5      06120      OR       L
2CFF CABE32  06130      JP       Z,NODRV     ;No drive entered
                06140 ;
                06150 ;      Get Start of Drive Code Table
                06160 ;
2D02 0E00    06170      LD       C,0        ;Get start of DCT
2D04          06180      @@GTDCT          ;Get DCT for Drive 0
2D04 3E51    060022     LD       A,81
2D06 EF      060023     RST      40
2D07 FDE5    06190      PUSH     IY        ;Get DCT start
2D09 D1      06200      POP      DE
                06210 ;
                06220 ;      Calculate Offset in Table
                06230 ;
2D0A AF      06240      XOR      A
2D0B ED52    06250      SBC      HL,DE      ;L = offset from start
2D0D B5      06260      OR       L          ;P/u offset
2D0E CAC232  06270      JP       Z,BADDRV   ;Cannot use DRIVE 0
                06280 ;
                06290 ;      Divide offset by 10 to get drive #
                06300 ;
2D11 06FF    06310      LD       B,-1       ;Init dividend = -1
2D13 04      06320      DIVLP1 INC      B        ;Bump dividend
2D14 D60A    06330      SUB      10        ;Subtract ten
2D16 30FB    06340      JR       NC,DIVLP1
                06350 ;
                06360 ;      Stuff away drive # into WRSEC routine
                06370 ;
2D18 78      06380      LD       A,B        ;P/u drive #
2D19 32C52F  06390      LD       (DRIVE+1),A ;Stuff away drive #
                06400 ;
                06410 ;      Point IY to System Flag table & RETurn
                06420 ;
2D1C          06430      @@FLAGS          ;IY => Flags

```

MEMDISKB - Subroutines

```

2D1C 3E65      00024      LD      A,101
2D1E EF        00025      RST      40
2D1F C9        06440      RET
                06450 ;
                06460 ; SAVDCT - Save Old DCT setup
                06470 ;
2D20 CDAE2C    06480 SAVDCT CALL    SAVEREG      ;Save registers
2D23 11003A    06490      LD      DE,DUPDCT    ;Destination
2D26 010A00    06500 DOXFER1 LD      BC,10      ;10 bytes to xfer
2D29 EDB0      06510      LDIR
2D2B C9        06520      RET
                06530 ;
                06540 ; GETDUP - Get Duplicate of original DCT setup
                06550 ;
2D2C ED5BBA32  06560 GETDUP LD      DE,(SAVEDCT) ;DE => DCT+0
2D30 21003A    06570      LD      HL,DUPDCT    ;Source
2D33 18F1      06580      JR      DOXFER1      ;Transfer back
                06590 ;
                06600 ; GTDRV - P/u Next available Driver Address
                06610 ;
                06620 ; IX <= Driver Address Pointer
                06630 ; DE <= Current Address
                06640 ;
2D35 E5        06650 GTDRV  PUSH    HL          ;Save HL
2D36 114B49    06660      LD      DE,'IK'      ;P/u *KI DCB address
2D39           06670      @@GTDCB
2D39 3E52      00026      LD      A,82
2D3B EF        00027      RST      40
2D3C 2B        06680      DEC     HL          ;KIDCB - 2 => free area
2D3D E5        06690      PUSH    HL          ;Xfer to IX
2D3E DDE1      06700      POP     IX
2D40 56        06710      LD      D,(HL)      ;P/u address in DE
2D41 2B        06720      DEC     HL
2D42 22F82E    06730      LD      (KIDCB$+1),HL ;Save address to stuff
2D45 5E        06740      LD      E,(HL)
2D46 E1        06750      POP     HL          ;Recover HL
2D47 C9        06760      RET
                06770 ;
                06780 ; INSTDRV - Relocate & Install Disk Driver
                06790 ;
2D48 EB        06800 INSTDRV EX      DE,HL      ;Xfer dest to HL
2D49 11BE2D    06810      LD      DE,DRIVER    ;Start of driver
2D4C E5        06820      PUSH    HL          ;Save Source & Dest ptrs
2D4D D5        06830      PUSH    DE
2D4E B7        06840      OR      A          ;Clear carry
2D4F ED52      06850      SBC     HL,DE      ;Get offset
                06860 ;
                06870 ; Relocate internal references in driver
                06880 ;
2D51 DD21702D  06890      LD      IX,RELTBL    ;Point to relocation tbl
2D55 44        06900      LD      B,H        ;Move to BC
2D56 4D        06910      LD      C,L
2D57 DD6E00    06920 RLOOP  LD      L,(IX)      ;Get address to change
2D5A DD6601    06930      LD      H,(IX+1)
2D5D 7C        06940      LD      A,H
2D5E B5        06950      OR      L
2D5F 2829      06960      JR      Z,RELDUN
2D61 5E        06970      LD      E,(HL)      ;P/U address
2D62 23        06980      INC     HL

```

MEMDISKB - Subroutines

```

2D63 56      06990      LD      D,(HL)
2D64 EB      07000      EX       DE,HL          ;Offset it
2D65 09      07010      ADD      HL,BC
2D66 EB      07020      EX       DE,HL
2D67 72      07030      LD       (HL),D          ;Put it back
2D68 2B      07040      DEC      HL
2D69 73      07050      LD       (HL),E
2D6A DD23    07060      INC      IX
2D6C DD23    07070      INC      IX
2D6E 18E7    07080      JR       RLOOP          ;Loop till done
          07090 ;
          07100 ;      Relocation Table for Driver
          07110 ;
2D70 002E    07120 RELTBL DW      REL1+1,REL2+1,REL3+1,REL4+1
          102E 412E
2D78 7A2E    07130      DW      REL5+1,REL6+2,REL7+1,REL8+1,REL8A+1
          E22D E62D EC2D 362E
2D82 7D2E    07140      DW      REL8B+1,REL9+1,REL2A+1,0
          E92D 072E 0000
          07150 ;
          07160 ;      Transfer MemDisk driver to driver area
          07170 ;
2D8A E1      07180 RELDUN POP      HL          ;HL => Source DE => Dest
2D8B D1      07190      POP      DE
2D8C D5      07200      PUSH     DE          ;Save start
2D8D 01DC00  07210      LD       BC,LENGTH ;# bytes to move
2D90 EDB0    07220      LDIR      ;Block move
2D92 D1      07230      POP      DE          ;Restore start
2D93 C9      07240      RET       ;Return
          07250 ;
          07260 ;      SETDCT - Set up Drive Code Table for MemDISK
          07270 ;
2D94 DD2ABA32 07280 SETDCT LD      IX,(SAVEDCT) ;IX => DCT address
2D98 DD3600C3 07290      LD      (IX+0),0C3H ;Enable
2D9C DD7301   07300      LD      (IX+1),E ;Lsb of driver
2D9F DD7202   07310      LD      (IX+2),D ;Msb of driver
2DA2 DD360340 07320 SDEND  LD      (IX+3),40H ;DD,5",floppy,step=6
2DA6 DD360450 07330 SDENE  LD      (IX+4),50H ;DDC=Y, 1 side, ALIEN
2DAA DD360500 07340      LD      (IX+5),0 ;Current Cyl = 0
2DAE DD7706   07350      LD      (IX+6),A ;# of tracks rel from 0
2DB1 DD360711 07360 SDENF  LD      (IX+7),17 ;18 spt (DD), 10 spt (SD)
2DB5 DD360845 07370 SDENG  LD      (IX+8),45H ;2/3 G/C, 5/6 S/G
2DB9 DD360901 07380      LD      (IX+9),1 ;Directory Cyl = 1
2DBD C9      07390      RET       ;Return
2DBE          00550 *GET  MEMDISK:3
          07400 ;MEMDISK/ASM - MemDISK Driver Code
2DBE          07410      SUBTTL  '<MEMDISK - MemDISK Driver>'

```

MEMDISKC - MemDISK Driver

```

2DBE          07420          PAGE
                07430 ;
2DBE 181D      07440 DRIVER JR      INIT          ;Jump around header
2DC0 0000      07450 OLDHIGH DW      0             ;Old HIDRV$
2DC2 03        07460          DB      3,'$MD'       ;Header
                24 4D 44
2DC6 0000      07470 OLD HI  DW      0             ;Old HIGH$ (for bank 0)
2DC8 00        07480 BANKIM DB      00000000B       ;Bank Image
2DC9 0000      07490 DRVLOW DW      0             ;What driver addr was
2DCB 0000      07500 MEMHIGH DW      0             ;HIGH$ after installed
                07510 ;
                07520          IF      @MOD2
                07530          DC      32,0          ;Model 2 stack area
                07540 ;
                07550          ELSE
2DCD 00        07560          DC      16,0          ;Driver Stack Area
                00 00 00 00 00 00 00 00
                00 00 00 00 00 00
                07570          ENDIF
2DDD          07580 MYSTACK EQU      $             ;Start of Mystack
                07590 ;
                07600 ;          Reset SP to MYSTACK, and CALL driver
                07610 ;
2DDD E5        07620 INIT      PUSH    HL           ;Save Registers
2DDE D5        07630          PUSH    DE           ;
2DDF C5        07640          PUSH    BC           ;
2DE0 ED73EF2D 07650 REL6      LD      (SAVE$P+1),SP ;Save original SP
2DE4 F3        07660          DI                   ;Interrupts off
2DE5 31DD2D    07670 REL7      LD      SP,MYSTACK   ;Memdisk SP
2DE8 228C2E    07680 REL9      LD      (BUFF+1),HL ;Save buffer addr request
2DEB CDF62D    07690 REL8      CALL    MEMDRIV     ;Call the actual driver
2DEE 310000    07700 SAVESP    LD      SP,$-$       ;P/u original SP
2DF1 FB        07710          EI                   ;Back on
2DF2 C1        07720          POP     BC           ;Restore Registers
2DF3 D1        07730          POP     DE
2DF4 E1        07740          POP     HL
2DF5 C9        07750          RET
                07760 ;
2DF6 78        07770 MEMDRIV LD      A,B           ;Get operation byte
                07780 ;
2DF7 FE09      07790 B9        CP      9           ;Operation #9 ?
2DF9 2027      07800          JR      NZ,B10        ;No - Check for Verify
                07810 ;
                07820 ;          READ sector - Set Z if D = directory cyl
                07830 ;
2DFB 15        07840          DEC     D             ;Set Z flag if Cyl = 1
2DFC F5        07850          PUSH    AF
2DFD 14        07860          INC     D             ;Restore cyl #
                07870 ;
                07880 ;          Set up For transfer to temporary I/O buffer
                07890 ;
2DFE E5        07900          PUSH    HL           ;Save User I/O buffer ptr
2DFF CD5D2E    07910 REL1      CALL    GETADR       ;HL => MemDISK Sector
2E02 3808      07920          JR      C,DOXFER      ;High - use temporary buf
                07930 ;
                07940 ;          I/O buff is low - xfer MemDISK sector to it
                07950 ;
2E04 EDB0      07960          LDIR          ;Xfer directly to buffer
2E06 CD832E    07970 REL2A     CALL    GETOLD       ;Get original bank
2E09 E1        07980          POP     HL           ;HL => User I/O buffer

```

MEMDISKC - MemDISK Driver

```

2E0A 180D      07990      JR      CHKDIR2      ;Check if directory cyl
                08000 ;
                08010 ;      Transfer MemDISK sector to Temporary Buffer
                08020 ;
2E0C D5        08030 DOXFER PUSH      DE      ;DE => Temporary Buffer
2E0D EDB0      08040 LDIR      ;Xfer to system area
                08050 ;
                08060 ;      Xfer data from temporary to User Buffer
                08070 ;
2E0F CD832E    08080 REL2      CALL      GETOLD      ;Get original bank
2E12 E1        08090          POP      HL      ;HL => Temporary buffer
2E13 D1        08100          POP      DE      ;DE => User I/O buffer
2E14 010001    08110          LD      BC,256      ;BC = 256 bytes to xfer
2E17 EDB0      08120          LDIR      ;Xfer to user buffer
                08130 ;
                08140 ;      Set A = Error #6 if Cylinder 1 (Directory)
                08150 ;
2E19 F1        08160 CHKDIR2 POP      AF      ;Get Z
2E1A 2004      08170 CHKDIR JR      NZ,NOTDIR      ;Not a directory read
2E1C 3E06      08180          LD      A,6      ;Error Code = 6
2E1E B7        08190          OR      A      ;NZ condition
2E1F C9        08200          RET      ;And RETURN
2E20 AF        08210 NOTDIR XOR      A      ;Set Z flag
2E21 C9        08220          RET      ;And return
                08230 ;
2E22 FE0A      08240 B10      CP      10      ;Verify sector ?
2E24 2003      08250 JR      NZ,B13      ;Check more if not
                08260 ;
                08270 ;      Verify a sector
                08280 ;
2E26 15        08290          DEC      D      ;Directory Cylinder
2E27 18F1      08300          JR      CHKDIR      ;Check if Directory cyl
                08310 ;
2E29 FE0D      08320 B13      CP      13      ;Write a sector?
2E2B 201E      08330 JR      NZ,B14      ;Check further if not
                08340 ;
                08350 ;      Write A Sector
                08360 ;
2E2D 3E0F      08370 WRITES LD      A,WP      ;WP error X'0F'
2E2F FDCB037E 08380          BIT      7,(IY+3)      ;Software Write Protect?
2E33 C0        08390          RET      NZ      ;Return with error
                08400 ;
                08410 ;      Set up for Tranfer to Temporary Buffer
                08420 ;
2E34 D5        08430          PUSH     DE      ;Save Cyl/Sector
2E35 CD8A2E    08440 REL8A    CALL     GETBUF      ;Get buffer ptr
2E38 3005      08450          JR      NC,RECVDE      ;Get back DE
2E3A 010001    08460          LD      BC,256      ;BC = 256 bytes to xfer
2E3D EDB0      08470          LDIR      ;Xfer to temp buffer
2E3F D1        08480 RECVDE POP      DE      ;DE = Cyl/sector
                08490 ;
                08500 ;      Get Sector from MemDISK & xfer to User buff
                08510 ;
2E40 CD5D2E    08520 REL3      CALL     GETADR      ;HL <= Mem, DE <= Buffer
2E43 EB        08530          EX      DE,HL
2E44 EDB0      08540          LDIR      ;Xfer to user buffer
2E46 CD832E    08550 REL4      CALL     GETOLD      ;Get original back
2E49 AF        08560          XOR      A      ;Set Z flag
2E4A C9        08570          RET

```

MEMDISKC - MemDISK Driver

```

08580 ;
2E4B FE0E 08590 B14 CP 14 ;Write system sector?
2E4D 28DE 08600 JR Z,WRITES ;Go if so
08610 ;
2E4F FE0C 08620 CP 12 ;Format command?
2E51 2804 08630 JR Z,B14A ;Go if so
2E53 FE0F 08640 CP 15 ;Write Track ?
2E55 2004 08650 JR NZ,EX1 ;No - exit Z
2E57 3E08 08660 B14A LD A,8 ;Yes - Exit NZ
2E59 B7 08670 OR A ;Error = Device not avail
2E5A C9 08680 RET
08690 ;
2E5B AF 08700 EX1 XOR A ;Zero A, set Z
2E5C C9 08710 RET ;Return with Z set
08720 ;
08730 ; GETADR - Point HL to MemDISK area
08740 ; - Point DE to Temporary buffer
08750 ; - Set BC = 256 (bytes to xfer)
08760 ;
2E5D 7A 08770 GETADR LD A,D ;P/u Cylinder #
08780 ;
08790 ; Multiply cylinder # x 10 or 18 (sectors/cyl)
08800 ;
2E5E 87 08810 SDENA ADD A,A ;X 2 or NOP if Single Den
2E5F 57 08820 LD D,A ;DDEN = x 2 SDEN = x 1
2E60 87 08830 ADD A,A ;DDEN = x 4 SDEN = x 2
2E61 87 08840 ADD A,A ;DDEN = x 8 SDEN = x 4
2E62 87 08850 SDENB ADD A,A ;DDEN = x 16 SDEN = x 5
2E63 82 08860 SDENC ADD A,D ;DDEN = x 18 SDEN = x 10
08870 ;
08880 ; Add Sect offset (E) & add 80H if bank 2 active
08890 ;
2E64 83 08900 ADD A,E ;Add sector offset
2E65 C600 08910 OFFSET ADD A,$-$ ;80H if 2 active
08920 ;
08930 ; Set HL => sector, C = Default bank (0 or 1)
08940 ;
2E67 67 08950 LD H,A ;Stuff msb in H
2E68 2E00 08960 LD L,0 ;Land on page boundary
2E6A 0E00 08970 DEFBANK LD C,$-$ ;C = 0 or C = 1
08980 ;
08990 ; Set C = Bank #2 if Address > X'7FFF'
09000 ;
2E6C 07 09010 RLCA ;Address > X'7FFF' ?
2E6D 3001 09020 JR NC,GOTBANK ;No - got it
2E6F 0C 09030 INC C ;Yes - Set C = 2
09040 ;
09050 ; Force address > X'7FFF' & Select Bank C
09060 ;
2E70 CBFC 09070 GOTBANK SET 7,H ;Force Address > X'7FFF'
2E72 45 09080 STFRET LD B,L ;Bring in Bank C
2E73 09090 @BANK
2E73 3E66 09028 LD A,102
2E75 EF 09029 RST 40
09100 ;
09110 ; Pick up Bank previously in use & Save
09120 ;
2E76 79 09130 LD A,C ;P/u last bank
2E77 E67F 09140 AND 7FH ;Ignore Hi-bit

```

MEMDISKC - MemDISK Driver

```

2E79 32842E 09150 REL5 LD (GETOLD+1),A ; and stuff away
          09160 ;
          09170 ; Set DE => Overlay Buffer, BC = 256
          09180 ;
2E7C CD8A2E 09190 REL8B CALL GETBUF ;Get buffer ptr
2E7F 010001 09200 LD BC,256 ;Set BC = 256
2E82 C9 09210 RET
          09220 ;
          09230 ; OLDBNK - Get original Bank used
          09240 ;
2E83 010000 09250 GETOLD LD BC,$-$ ;B = 0, C = Bank #
2E86 09260 @BANK ;Get bank
2E86 3E66 09270 LD A,102
2E88 EF 09280 RST 40
2E89 C9 09290 RET
          09280 ;
          09290 ; GETBUF - Get Buffer ptr to LDIR from or to
          09300 ;
          09310 ;
2E8A E5 09310 GETBUF PUSH HL ;Save source/dest ptr
2E8B 110000 09320 BUFF LD DE,$-$ ;P/u requested I/O buffer
2E8E 21007F 09330 LD HL,7F00H ;Use (BUFF+1) if < 7F00H
2E91 B7 09340 OR A
2E92 ED52 09350 SBC HL,DE ;Past 7F00H ?
2E94 E1 09360 POP HL ;Rcvr ptr
2E95 D0 09370 RET NC ;No - use requested buff
2E96 110023 09380 LD DE,BUFFER$ ;Yes - use BUFFER$
2E99 C9 09390 RET
          09400 ;
00DC 09410 LENGTH EQU $-DRIVER ;Length of Driver
2E9A 00560 *GET MEMDISKA:3
          09420 ;MEMDISKA/ASM - Memdisk Initialization
2E9A 09430 SUBTTL '<MEMDISKA - Installation>'

```

MEMDISKA - Installation

```

2E9A          09440      PAGE
                09450 ;
2E9A F5       09460 INSTMEM PUSH AF          ;Save # cyls
2E9B C5       09470      PUSH BC          ;Save Bank #
                09480 ;
                09490 ;      Is there a MemDISK driver trapped ?
                09500 ;
2E9C 11C034   09510      LD DE,MD$          ;"$MD"
2E9F          09520      @@GTMOD          ;MemDISK in ?
2E9F 3E53     09530      LD A,83
2EA1 EF       09533      RST 40
2EA2 2011     09530      JR NZ,NOT__IN      ;No
                09540 ;
                09550 ;      There is a driver trapped - use that area
                09560 ;
2EA4 22EF2E   09570      LD (OLDRVR+1),HL    ;Save old driver addr
2EA7 EB       09580      EX DE,HL          ;Pt DE => Destination
2EA8 216737   09590      LD HL,RE_USE      ;Set re-use flag
2EAB 34       09600      INC (HL)
2EAC 21DB00   09610      LD HL,LENGTH-1     ;Set HL = last used
2EAF 19       09620      ADD HL,DE          ; address of driver
2EB0 22C02D   09630      LD (OLDHIGH),HL    ;Xfer into driver
2EB3 1827     09640      JR DO_INST         ;Install driver
                09650 ;
                09660 ;      Driver is not in memory - is there room ?
                09670 ;
2EB5 CD352D   09680 NOT__IN CALL GTDRV      ;P/u low driver ptr
2EB8 ED53EF2E 09690      LD (OLDRVR+1),DE    ;Save it
2EBC 21DB00   09700      LD HL,LENGTH-1     ;HL = length of driver
2EBF 010013   09710      LD BC,HIDRVR      ;BC = 1 + highest avail
2EC2 19       09720      ADD HL,DE          ;HL => Last used by Mem
2EC3 22C02D   09730      LD (OLDHIGH),HL
2EC6 23       09740      INC HL
2EC7 B7       09750      OR A
2EC8 E5       09760      PUSH HL           ;Will MemDisk fit ?
2EC9 ED42     09770      SBC HL,BC
2ECB E1       09780      POP HL
2ECC 3808     09790      JR C,OKTOGO       ;Yes - let's do it
                09800 ;
                09810 ;      Insufficient Driver space
                09820 ;
2ECE 21D232   09830      LD HL,NOMEM       ;Alter exit message
2ED1 22112F   09840      LD ($NOT+1),HL
2ED4 1818     09850      JR OLDRVR         ;Reclaim hi mem if bank 0
                09860 ;
                09870 ;      Save next avail mem addr & set Memdisk bit
                09880 ;
2ED6 DD7400   09890 OKTOGO LD (IX),H        ;Stuff msb
2ED9 DD75FF   09900      LD (IX-1),L        ;Stuff lsb
                09910 ;
                09920 ;      Install MemDISK driver & set up DCT
                09930 ;
2EDC CD482D   09940 DO_INST CALL INSTDRV    ;Relocate, install driver
2EDF C1       09950      POP BC            ;C = Bank # requests
2EE0 F1       09960      POP AF            ;A = # cylinders
2EE1 CD942D   09970      CALL SETDCT       ;Set up DCT
                09980 ;
                09990 ;      Prompt for Format
                10000 ;
                10010 ;
2EE4 CD7C32   10010      CALL FORMTIT      ;Format this ?

```


MEMDISKA - Installation

```

2EE7 282A    10020    JR      Z,DOFORM1      ;Yes - do it
              10030    ;
              10040    ;      Format = No, Is there a MemDISK here ?
              10050    ;
2EE9 3E00    10060    MEMIN1 LD      A,$-$      ;0 = not active
2EEB B7      10070    OR      A              ;
2EEC 2034    10080    JR      NZ,SHOWINU      ;MemDisk previously in
              10090    ;
              10100    ;      Abort installation - stuff X'C9' in DCT
              10110    ;
2EEE 210000  10120    OLDVR  LD      HL,$-$      ;P/u original driver addr
2EF1 3A6737  10130    LD      A,(RE_USE)      ;Have we re-used driver
2EF4 B7      10140    OR      A              ; area that was trapped ?
2EF5 2003    10150    JR      NZ,DONTRES      ;Yes - don't reset memptr
2EF7 220000  10160    KIDCB$ LD      ($-$),HL    ;Stuff ptr used
2EFA 2ABA32  10170    DONTRES LD      HL,(SAVEDCT) ;P/u DCT address
2EFD 36C9    10180    LD      (HL),0C9H      ;Disable it
2EFF FDCB03A6 10190    RES     4,(IY+DFLAG$)    ;Reset MemDISK bit
2F03 3A292C  10200    LD      A,(SETBANK+1)    ;P/u bank request
2F06 B7      10210    OR      A              ;If alternate bank(s),
2F07 2007    10220    JR      NZ,$NOT        ; don't reset high$
2F09 2AC634  10230    LD      HL,(MDDATA+2)    ;Pu old high$
2F0C 47      10240    LD      B,A
2F0D        10250    @@HIGH$                ;Reset high$
2F0D 3E64    00034    LD      A,100
2F0F EF      00035    RST     40
2F10 C3DE32  10260    $NOT   JP      NOTACT      ;Show not installed
              10270    ;
              10280    ;      Format mem, init GAT & HIT, & BOOT-DIR entries
              10290    ;
2F13 CD2D32  10300    DOFORM1 CALL    FORMAT      ;Format
2F16 CD302F  10310    CALL    WRBOOT      ;Write BOOT/SYS
2F19 CD5A2F  10320    CALL    WRGAT      ;Initialize GAT
2F1C CDD52F  10330    CALL    WRHIT      ;Initialize HIT
2F1F CDE22F  10340    CALL    WRENT      ;Put DIR & BOOT entries
2F22 CD282C  10350    SHOWINU CALL    SETBANK    ;Show Banks in use
2F25 FDCB03E6 10360    SET     4,(IY+DFLAG$)    ;Set MemDisk flag
2F29 211335  10370    LD      HL,INSTALD      ;Init"MemDisk Installed
2F2C        10380    @@LOGOT                ;Display the msg
              00036    IFEQ    00H,1
              00037    LD      HL,
              00038    ENDIF
2F2C 3E0C    00039    LD      A,12
2F2E EF      00040    RST     40
2F2F C9      10390    RET                    ;Done - GO TO EXIT
              10400    ;
              10410    ;      WRBOOT - Write BOOT/SYS information
              10420    ;
2F30 AF      10430    WRBOOT XOR     A          ;Fill byte
2F31 210038  10440    LD      HL,IOBUFF      ;HL => I/O buffer
              10450    ;
              10460    ;      Fill BOOT/SYS with Zeroes
              10470    ;
2F34 77      10480    FILBUF LD      (HL),A      ;Stuff in byte
2F35 2C      10490    INC     L              ;One sector to
2F36 20FC    10500    JR      NZ,FILBUF      ; fill
              10510    ;
              10520    ;      Write # of Sectors in BOOT
              10530    ;

```

MEMDISKA - Installation

```

2F38 57      10540      LD      D,A          ;Cylinder 0
2F39 5F      10550      LD      E,A          ;Sector 0
2F3A 0606    10560 BTSECS LD      B,6          ;P/u Sec cnt - 5,6, or 18
2F3C CDC12F  10570 BTLP  CALL     WRSEC        ;Write sector
2F3F 1C      10580      INC      E          ;Bump
2F40 10FA    10590      DJNZ     BTLP
                10600 ;
                10610 ;      Write Directory Cylinder byte in Sector Zero
                10620 ;
                10630 ;
2F42 2E02    10630      LD      L,2          ;Byte 2
2F44 3601    10640      LD      (HL),1        ;Directory cyl = 1
                10650 ;
                10660 ;      Write Sector 0 of Cylinder 0
                10670 ;
2F46 110000  10680      LD      DE,0          ;Cylinder 0, Sector 0
2F49 CDC12F  10690      CALL     WRSEC        ;Write Sector
                10700 ;
                10710 ;      Make a duplicate of sector 0 in sector 1
                10720 ;
2F4C 1C      10730      INC      E          ;Sector 1
2F4D CDC12F  10740      CALL     WRSEC        ;Write sector
                10750 ;
                10760 ;      Write C/R in Auto Buffer in Sector 2
                10770 ;
2F50 1E02    10780      LD      E,2          ;Sector 2
2F52 2E20    10790      LD      L,20H        ;Byte X'20'
2F54 360D    10800      LD      (HL),CR      ;No auto
2F56 CDC12F  10810      CALL     WRSEC        ;Write sector
2F59 C9      10820      RET              ;RETurn for now
                10830 ;
                10840 ;      WRGAT - Write Granule Allocation Table
                10850 ;
                10860 ;
2F5A 210038  10870 WRGAT LD      HL,IOBUFF      ;HL => I/O buffer
2F5D 36F9    10880 GAT0  LD      (HL),0F9H      ;DD - X'F9', SD - X'FD'
2F5F 23      10890      INC      HL          ;Bump
                10900 ;
                10910 ;      Lock out next X'CA' bytes in GAT
                10920 ;
2F60 06CA    10930      LD      B,0CAH        ;Lock out the bytes
2F62 36FF    10940 LOCKOUT LD      (HL),0FFH      ;GAT + X'01' through
2F64 23      10950      INC      HL          ;GAT + X'CA'
2F65 10FB    10960      DJNZ     LOCKOUT
                10970 ;
                10980 ;      GAT + X'CB'
                10990 ;
2F67 3662    11000      LD      (HL),62H        ;GAT + X'CB' = Version 6.2
                11010 ;
                11020 ;      GAT + X'CC'
                11030 ;
2F69 3E00    11040 CYLS LD      A,$-$        ;P/u cylinder count
2F6B F5      11050      PUSH     AF          ;Save Cylinder count
2F6C D623    11060      SUB      35          ;Tracks in excess of 35
2F6E 23      11070      INC      HL          ;HL => next GAT byte
2F6F 77      11080      LD      (HL),A        ;GAT + X'CC' = tracks - 35
                11090 ;
                11100 ;      GAT + X'CD'
                11110 ;
2F70 23      11120      INC      HL          ;GAT + X'CD' =

```

MEMDISKA - Installation

```

2F71 3642      11130 GATCD   LD      (HL),42H      ;DDEN, 1 side, 3 gran/cyl
                11140 ;
                11150 ;      GAT + X'CE' & X'CF'
                11160 ;

2F73 23        11170      INC      HL              ;GAT + X'CE' & X'CF' =
2F74 36E0      11180      LD      (HL),0E0H        ;16-bit Hash code of
2F76 23        11190      INC      HL              ;"PASSWORD"
2F77 3642      11200      LD      (HL),42H        ;Hash = X'42E0'
                11210 ;
                11220 ;      GAT + X'D0' - X'D7'
                11230 ;

2F79 23        11240      INC      HL              ;HL => next GAT byte
2F7A 11B834    11250      LD      DE,MEMDISK      ;"MEMDISK " is Pack name
2F7D 0E08      11260      LD      C,8             ;Eight bytes
2F7F EB        11270      EX       DE,HL          ;Swap 'em for LDIR
2F80 EDB0      11280      LDIR                     ;Stuff in ID
2F82 EB        11290      EX       DE,HL          ;HL => GAT + X'D8'
                11300 ;
                11310 ;      GAT + X'D8' - X'DF'
                11320 ;

2F83           11330      @@DATE                  ;Stuff date in GAT
2F83 3E12      00041      LD      A,18
2F85 EF        00042      RST      40
                11340 ;
                11350 ;      Stuff GAT tracks in use with either X'F8' or X'FC'
                11360 ;

2F86 3EF8      11370 GPC   LD      A,0F8H          ;3 gran/cyl
2F88 210238    11380      LD      HL,IOBUFF+2      ;HL => GAT + X'02'
2F8B C1        11390      POP      BC              ;B = # cylinders
2F8C 05        11400      DEC      B              ;Subtract 2 to account
2F8D 05        11410      DEC      B              ;For BOOT and DIR
                11420 ;
                11430 ;      Stuff open cylinder bytes into GAT
                11440 ;

2F8E 77        11450 FREETRK LD      (HL),A        ;Free track
2F8F 23        11460      INC      HL              ;Next GAT byte
2F90 10FC      11470      DJNZ     FREETRK         ;Do it B times
                11480 ;
                11490 ;      Put 2 free Cyl bytes in lockout - BOOT & DIR
                11500 ;

2F92 2E60      11510      LD      L,60H           ;HL => Lockout
2F94 77        11520      LD      (HL),A
2F95 2C        11530      INC      L
2F96 77        11540      LD      (HL),A
                11550 ;
                11560 ;      GAT + X'62' - GAT + X'BF'
                11570 ;

2F97 2E02      11580      LD      L,2             ;HL => GAT + X'02'
2F99 54        11590      LD      D,H             ;Xfer to DE
2F9A 5D        11600      LD      E,L
2F9B 0E60      11610      LD      C,60H           ;Of X'60' for the
2F9D 09        11620      ADD      HL,BC          ; duplicate of top
2F9E 0D        11630      DEC      C              ;Only duplicate X'5E'
2F9F 0D        11640      DEC      C              ; bytes
2FA0 EB        11650      EX       DE,HL          ;Prepare for LDIR
2FA1 EDB0      11660      LDIR                     ;HL => GAT, DE => Lockout
                11670 ;

2FA3 11F438    11680      LD      DE,IOBUFF+255-11 ;6.2 Media Data Block
2FA6 21BA2F    11690      LD      HL,LSIID        ;Point to header

```

MEMDISKA - Installation

```

2FA9 010400 11700 LD BC,04 ;Set length
2FAC EDB0 11710 LDIR ;Move it
2FAE 2ABA32 11720 LD HL,(SAVEDCT) ;The data to move
2FB1 23 11730 INC HL
2FB2 23 11740 INC HL
2FB3 23 11750 INC HL
2FB4 0E07 11760 LD C,7 ;Bytes to move
2FB6 EDB0 11770 LDIR ;Move it in
2FB8 1804 11780 JR WRGAT1 ;Skip around string
2FBA 03 11790 LSIID DB 03,'LSI'
    4C 53 49

2FBE 110001 11800 ;
11810 WRGAT1 LD DE,100H ;D = Cyl 1, E = Sector 0
11820 ;
11830 ; WRSEC - Write A sector to MemDISK drive
11840 ;
2FC1 210038 11850 WRSEC LD HL,IOBUFF ;I/O buffer
2FC4 0E00 11860 DRIVE LD C,$-$ ;P/u drive #
2FC6 11870 @WRSEC ;Write Sector
2FC6 3E35 00043 LD A,53
2FC8 EF 00044 RST 40
2FC9 C9 11880 RET ; and RETurn
11890 ;
11900 ; RDSEC - Read A sector of MemDISK drive
11910 ;
2FCA 210038 11920 RDSEC LD HL,IOBUFF ;HL => I/O Buffer
2FCD 3AC52F 11930 LD A,(DRIVE+1) ;P/u drive #
2FD0 4F 11940 LD C,A ;Xfer to C
2FD1 11950 @RDSEC ;Read sector
2FD1 3E31 00045 LD A,49
2FD3 EF 00046 RST 40
2FD4 C9 11960 RET ; and RETurn
11970 ;
11980 ; WRHIT - Write HIT sector in directory
11990 ;
2FD5 AF 12000 WRHIT XOR A ;Set A = 0
2FD6 77 12010 ZEROHIT LD (HL),A ;Zero HIT position
2FD7 2C 12020 INC L ;Bump HIT pointer
2FD8 20FC 12030 JR NZ,ZEROHIT ;256 positions
2FDA 36A2 12040 LD (HL),0A2H ;Hash for BOOT/SYS
2FDC 2C 12050 INC L ;HL => HIT + X'01'
2FDD 36C4 12060 LD (HL),0C4H ;Hash for DIR/SYS
2FDF 1C 12070 INC E ;D = Cyl 1, Sector 1
2FE0 18DF 12080 JR WRSEC ;Write Sector & RETurn
12090 ;
12100 ; WRENT - Write DIR/SYS & BOOT/SYS entries
12110 ;
2FE2 11FF2F 12120 WRENT LD DE,BOOT ;BOOT/SYS byte field
2FE5 EB 12130 EX DE,HL ;Swap for LDIR
2FE6 012000 12140 LD BC,32 ;32 bytes in entry
2FE9 EDB0 12150 LDIR ;Block move
2FEB 110201 12160 LD DE,102H ;D = Cyl 1, E = Sector 2
2FEE CDC12F 12170 CALL WRSEC ;Write Sector
12180 ;
2FF1 012000 12190 LD BC,32
2FF4 EB 12200 EX DE,HL ;Xfer buffer ptr to DE
2FF5 211F30 12210 LD HL,DIR ;HL => DIR/SYS bytes
2FF8 EDB0 12220 LDIR ;Xfer to MemDISK
2FFA 110301 12230 LD DE,103H ;D = Cyl 1, E = Sector 3

```

MEMDISKA - Installation

```

2FFD 18C2      12240      JR      WRSEC      ;Write sector & RETURN
                12250 ;
                12260 ;      BOOT/SYS directory entry data
                12270 ;
2FFF 5E        12280 BOOT  DB      01011110B    ;No access,inv,sys,FPDE
3000 0000      12290      DW      0           ;Date = 00/00/00
3002 0000      12300      DW      0           ;EOF offset = 0, LRL=256
3004 42        12310      DB      'BOOT'      ;Name field
                4F 4F 54 20 20 20 20
300C 53        12320      DB      'SYS'        ;Extension
                59 53
300F F637      12330      DW      037F6H      ;Owner password hash
3011 F59C      12340      DW      09CF5H      ;User password hash
3013 0600      12350 BOOTERN DW      6         ;ERN = 6 or 5
3015 00        12360      DB      0           ;First extent = Cyl 0
3016 00        12370 BOOTGRN DB      0         ;St gran = 0, 1 cont gran
3017 FFFF      12380      DW      0FFFFH      ;No more extents
3019 FFFF      12390      DW      0FFFFH
301B FFFF      12400      DW      0FFFFH
301D FFFF      12410      DW      0FFFFH
                12420 ;
                12430 ;      DIR/SYS directory entry data
                12440 ;
301F 5D        12450 DIR   DB      01011101B    ;Read only,inv,sys,FPDE
3020 0000      12460      DW      0           ;Date= 00/00/00
3022 0000      12470      DW      0           ;EOF offset=0, LRL=256
3024 44        12480      DB      'DIR'       ;Name field
                49 52 20 20 20 20 20
302C 53        12490      DB      'SYS'        ;Extension
                59 53
302F F637      12500      DW      037F6H      ;Owner password hash
3031 9642      12510      DW      04296H      ;User password hash
3033 1200      12520 DIRERN DW      18        ;ERN+1 = 10 or 18
3035 01        12530      DB      1           ;Starts on cylinder 1
3036 02        12540 SDENI DB      00000010B    ;St. gran=0, 3 cont grans
3037 FFFF      12550      DW      0FFFFH      ;No Second Extent
3039 FFFF      12560      DW      0FFFFH      ;No Third Extent
303B FFFF      12570      DW      0FFFFH      ;No Fourth Extent
303D FF        12580      DB      0FFH        ;No further records
303E FF        12590      DB      0FFH
                12600 ;
                12610 ;      DOMEM - Issue Prompts & take inputs for type
                12620 ;
303F 21E732    12630 DOMEM LD      HL,HELLO$    ;Display message
3042          12640      @@DSPLY
                00047      IFEQ      00H,1
                00048      LD        HL,
                00049      ENDIF
3042 3E0A      00050      LD        A,10
3044 EF        00051      RST      40
                12650 ;
                12660 ;      Check if entry from SYSTEM (DRIVER= command
                12670 ;
                12680      @@FLAGS
3045          12680      LD        A,101
3045 3E65      00052      RST      40
3047 EF        00053      BIT      3,(IY+'C'-'A') ;System request?
3048 FDCB025E  12690      JP      Z,VIASET    ;Quit if not
304C CAC632    12700
                12710 ;

```

MEMDISKA - Installation

```

12720 ;      Input MemDISK type - A,B,C,D or E to disable
12730 ;
304F 216D33 12740 GETYPE LD      HL,BANKS      ;Display prompt
3052        12750 @0DSPLY
          00054 IFEQ    00H,1
          00055 LD      HL,
          00056 ENDF
3052 3E0A   00057 LD      A,10
3054 EF     00058 RST      40
3055 0601   12760 LD      B,1      ;# of chars to input
3057 CDD62C 12770 CALL     INPUT      ;Input byte
305A 28F3   12780 JR      Z,GETYPE    ;<ENTER> ? - re-input
          12790 ;
          12800 ;      Convert input A-E to 0-4
          12810 ;
305C 7E     12820 LD      A,(HL)      ;P/u first character
305D CBAF   12830 RES      5,A      ;Convert to U/C
305F D641   12840 SUB      'A'      ;<A> - Bank 0 ?
3061 32292C 12850 LD      (SETBANK+1),A ;Save type of MemDISK
3064 4F     12860 LD      C,A      ;Xfer to C for @BANK
          12870 ;
          12880 ;      If input is illegal then re-input
          12890 ;
3065 38E8   12900 JR      C,GETYPE    ;Less - re-input
3067 FE04   12910 CP      4      ;<E> - Disable MemDISK
3069 CA8A31 12920 JP      Z,DISMEM    ;Yes - take it out
306C 30E1   12930 JR      NC,GETYPE   ;>4 - Re-input
          12940 ;
          12950 ;      Check if MemDISK is already active
          12960 ;
306E FDCB0366 12970 BIT      4,(IY+DFLAG$) ;MemDISK already active ?
3072 C2CA32  12980 JP      NZ,MEMIN    ;Yes - abort
          12990 ;
          13000 ;      If Type A,B,C - Check Bk, D - Check bks 1&2
          13010 ;
3075 C5     13020 PUSH     BC      ;Save Bank #
3076 FE03   13030 CP      3      ;Type "D" ?
3078 2006   13040 JR      NZ,A_B_C    ;No - "A", "B", or "C"
          13050 ;
          13060 ;      Type "D" - See if both banks 1 & 2 are avail
          13070 ;
307A 0E01   13080 TYPED LD      C,1      ;Bank #1 active ?
307C CDCB2C 13090 CALL     CKBANK
307F 0C     13100 INC      C      ;Bank #2 active ?
3080 CDCB2C 13110 A_B_C CALL     CKBANK
3083 C1     13120 POP      BC      ;C = Bank # (0,1,2,3)
          13130 ;
          13140 ;      Stuff Default Bank # and offset into driver
          13150 ;
3084 79     13160 LD      A,C      ;P/u bank #
3085 3D     13170 DEC      A      ;If bank 0 requested,
3086 FA9830 13180 JP      M,WAS0    ; then keep as -1
3089 3C     13190 INC      A      ; for driver bank test
308A 32C82D 13200 LD      (BANKIM),A ;Save bank # in driver
308D FE02   13210 CP      2      ;Instruction if
308F 2005   13220 JR      NZ,NOT2    ;Just bank #2 active
3091 21662E 13230 LD      HL,OFFSET+1 ;Stuff X'80' in ADD
3094 3680   13240 LD      (HL),80H
3096 3E01   13250 NOT2   LD      A,1      ;Always init to bank 1

```

MEMDISKA - Installation

```

13260
3098 326B2E 13270 WAS0 LD (DEFBANK+1),A ; if type B, C or D
13280 ; ; Stuff in driver
13290 ; ; Input Density (Single or Double)
13300 ;
309B 215234 13310 INPDENS LD HL,DENSITY ;"Density"
309E 13320 @@DSPLY
00059 IFEQ 00H,1
00060 LD HL,
00061 ENDIF
309E 3E0A 00062 LD A,10
30A0 EF 00063 RST 40
30A1 0601 13330 LD B,1 ;Input an "S" or "D"
30A3 CDD62C 13340 CALL INPUT
30A6 2856 13350 JR Z,DEFAULT ;<ENTER> - use default
13360 ;
13370 ; <D>ouble Density input ?
13380 ;
30A8 7E 13390 LD A,(HL) ;P/u first char
30A9 CBAF 13400 RES 5,A ;Convert to U/C
30AB FE44 13410 CP 'D' ;<D>ouble Density ?
30AD 284F 13420 JR Z,DEFAULT ;Yes - use 6 sectors/gran
13430 ;
13440 ; <S>ingle Density input ?
13450 ;
30AF FE53 13460 CP 'S' ;<S>ingle Density ?
30B1 20E8 13470 JR NZ,INPDENS ;No - input density again
13480 ;
13490 ; Single Density - Change driver math
13500 ;
30B3 3E82 13510 LD A,82H ;ADD A,D instruction
30B5 32622E 13520 LD (SDENB),A
30B8 3E87 13530 LD A,87H ;ADD A,A instruction
30BA 32632E 13540 LD (SDENC),A
30BD 3E09 13550 LD A,9
30BF 32B42D 13560 LD (SDENF+3),A ;DCT + 7
30C2 326A31 13570 LD (SPC+1),A ;Save in CALCSIZ routine
30C5 3C 13580 INC A ;SDEN BOOT ERN = 10
30C6 323330 13590 LD (DIRERN),A ;SDEN DIR/SYS ERN = 10
30C9 3E24 13600 LD A,24H
30CB 32B82D 13610 LD (SDENG+3),A ;DCT + 8
30CE 3E32 13620 LD A,'2' ;Change size to 2.50K
30D0 322034 13630 LD (FRTRK1),A ;Space per cylinder
30D3 3EFD 13640 LD A,0FDH ;1 Gran Free
30D5 325E2F 13650 LD (GAT0+1),A ;Stuff in WRGAT routine
30D8 3D 13660 DEC A ;2 Grans/Cyl - X'FC'
30D9 32872F 13670 LD (GPC+1),A
30DC AF 13680 XOR A ;NOP instruction
30DD 325E2E 13690 LD (SDENA),A
30E0 32A52D 13700 LD (SDEND+3),A ;DCT + 3
30E3 3C 13710 INC A ;Set A = 1
30E4 32722F 13720 LD (GATCD+1),A ;Stuff in WRGAT routine
30E7 323630 13730 LD (SDENI),A ;2 contiguous granules
30EA 3E05 13740 LD A,5 ;Set Boot ERN = 5
30EC 321330 13750 LD (BOOTERN),A
30EF 3E10 13760 LD A,10H ;Alien Disk Controller
30F1 32A92D 13770 LD (SDENE+3),A
30F4 213B2F 13780 LD HL,BTSECS+1 ;HL => # BOOT sectors
30F7 35 13790 DEC (HL) ;Use 5 instead of 6

```

MEMDISKA - Installation

```

30F8 21000A 13800 LD HL,SDBPC ;Change GETCYL routine
30FB 22E52C 13810 LD (BPC+1),HL
13820 ;
13830 ; Calculate # of possible cylinders
13840 ;
30FE 3A292C 13850 DEFAULT LD A,(SETBANK+1) ;P/u type of memdisk
3101 4F 13860 LD C,A ;Save in C
3102 B7 13870 OR A ;Bank 0 ?
3103 280A 13880 JR Z,PIKUPHI ;Yes - use HIGH$
13890 ;
13900 ; Bank #1, #2, or #1 & #2
13910 ;
3105 21FF7F 13920 LD HL,7FFFH ;HL = # bytes in 1 bank
3108 FE03 13930 CP 3 ;Bank 1 & 2 ?
310A 201F 13940 JR NZ,CALCYL ;No - use X'7FFF'
310C 65 13950 LD H,L ;Set HL = X'FFFF'
310D 181C 13960 JR CALCYL
13970 ;
13980 ; Bank Zero request - calculate free mem avail
13990 ;
310F AF 14000 PIKUPHI XOR A ;Set A = 0
3110 ED62 14010 SBC HL,HL ;HL = 0
3112 47 14020 LD B,A ;B = 0
3113 14030 @@HIGH$ ;P/u HIGH$
3113 3E64 00064 LD A,100
3115 EF 00065 RST 40
3116 22C634 14040 LD (MDDATA+2),HL ;Save HIGH$
3119 22C62D 14050 LD (OLD_HI),HL ;Save HIGH$ in driver
311C 23 14060 INC HL ;Set HL = last page
311D 25 14070 DEC H
311E 6F 14080 LD L,A
311F 226F31 14090 LD (SAVPAGE+1),HL ;Save page boundary
3122 110080 14100 LD DE,LOWEST ;DE = lowest
3125 AF 14110 XOR A
3126 ED52 14120 SBC HL,DE ;HL = amount free
3128 DAD232 14130 JP C,NOMEM ;Carry - not enough mem
14140 ;
14150 ; Calculate # of cylinders available
14160 ;
312B CDE22C 14170 CALCYL CALL GETCYL ;Get # of poss cyls
312E C2D232 14180 JP NZ,NOMEM ;NZ - Not enough mem
14190 ;
14200 ; Convert A to ASCII & stuff into string
14210 ;
3131 3C 14220 INC A ;Bump one
3132 325F31 14230 LD (MAXCYL+1),A ;Save max # of cyls
3135 3D 14240 DEC A
3136 326A2F 14250 LD (CYLS+1),A ;Stuff in WRGAT routine
3139 F5 14260 PUSH AF ;Save Max # of cyls
313A CD632C 14270 CALL DECASC ;Convert to ASCII in HL
313D F1 14280 POP AF ;A = # cyls
313E EB 14290 EX DE,HL ;DE = #
313F 214C34 14300 LD HL,FRTRK2 ;HL => Destination
3142 72 14310 LD (HL),D ;Msb
3143 23 14320 INC HL
3144 73 14330 LD (HL),E ;Lsb
14340 ;
14350 ; A = # of Cyls poss, put in string if bank 0
14360 ;

```


MEMDISKA - Installation

```

3145 0C      14370      INC      C              ;Bank Zero request ?
3146 0D      14380      DEC      C
3147 C0      14390      RET      NZ              ;No - done prompting
          14400 ;
          14410 ;      Display Cylinders string & input # of cyls
          14420 ;
3148 210534  14430 REDO   LD      HL,FRTRACK      ;How many cylinders
314B         14440      @@DSPLY
          00066      IFEQ    00H,1
          00067      LD      HL,
          00068      ENDIF
314B 3E0A    00069      LD      A,10
314D EF      00070      RST      40
314E 0602    14450      LD      B,2              ;Input # of cyls
3150 CDD62C  14460      CALL    INPUT
3153 28F3    14470      JR      Z,REDO           ;Reinput it
          14480 ;
          14490 ;      Check if input legal
          14500 ;
3155 CD6E2C  14510      CALL    DECHEX          ;Convert # to Hex
3158 20EE    14520      JR      NZ,REDO         ;Illegal - Re-input
315A FE03    14530      CP      MINCYL          ;Less than minimum?
315C 38EA    14540      JR      C,REDO
315E FE00    14550 MAXCYL CP      $-$          ;P/u max # of cyls
3160 30E6    14560      JR      NC,REDO         ;Too many - reinput
3162 326A2F  14570      LD      (CYLS+1),A      ;New # of cylinders
          14580 ;
          14590 ;      CALCSIZ - Calculate Size of Cyl request
          14600 ;
3165 CDAE2C  14610 CALCSIZ CALL    SAVEREG        ;Save Registers
3168 4F      14620      LD      C,A            ;Xfer # cyls to C
3169 0611    14630 SPC    LD      B,17          ;P/u Sectors/Cyl
          14640 ;
          14650 ;      Multiply Sectors per Cylinder x # Cylinders
          14660 ;
316B 81      14670 MLOOP  ADD     A,C            ;Multiply B x C
316C 10FD    14680      DJNZ    MLOOP
          14690 ;
          14700 ;      Set HL = New HIGH$
          14710 ;
316E 210000  14720 SAVPAGE LD      HL,$-$        ;P/u page boundary
3171 ED44    14730      NEG                     ;Set H = H - A
3173 84      14740      ADD     A,H
3174 67      14750      LD      H,A            ;HL = New HIGH$, B = 0
3175 32662E  14760      LD      (OFFSET+1),A    ;Stuff into driver
          14770 ;
          14780 ;      Stuff a Memory Header on front of MemDISK
          14790 ;
3178 2B      14800      DEC     HL              ;Pt 1 byte before
3179 EB      14810      EX      DE,HL          ; Memdisk himem area
317A 21D434  14820      LD      HL,MDDATA+16    ;Pt to header block
317D 011100  14830      LD      BC,17
3180 EDB8    14840      LDDR                     ; and move it to himem
3182 EB      14850      EX      DE,HL
3183 22CB2D  14860      LD      (MEMHIGH),HL
3186         14870      @@HIGH$
          00071      LD      A,100
3188 EF      00072      RST      40
3189 C9      14880      RET                     ;Restore Regs & RETURN

```

MEMDISKA - Installation

```

14890 ;
14900 ; DISMEM - Disable MemDISK if in memory
14910 ;
318A FDCB0366 14920 DISMEM BIT 4,(IY+DFLAG$) ;MemDISK active ?
318E CAD632 14930 JP Z,NOTPRS ;No - display error mess
14940 ;
14950 ; Pick up Driver address of drive
14960 ;
3191 2ABA32 14970 LD HL,(SAVEDCT) ;P/u DCT address
3194 E5 14980 PUSH HL ;Save DCT ptr
3195 23 14990 INC HL ;P/u driver address
3196 5E 15000 LD E,(HL) ;Lsb
3197 23 15010 INC HL
3198 56 15020 LD D,(HL) ;Msb
3199 D5 15030 PUSH DE ;Save Driver Address
15040 ;
15050 ; Calculate end of driver & Posn to ID
15060 ;
319A EB 15070 EX DE,HL ;Pt HL to driver
319B E5 15080 PUSH HL ;Save driver start
319C 01DC00 15090 LD BC,LENGTH ;Add length of driver
319F 09 15100 ADD HL,BC ; to start of driver.
31A0 22F831 15110 LD (DREND+1),HL ;Save next available
31A3 E1 15120 POP HL ;HL => driver add start
31A4 23 15130 INC HL ;Pos'n to length byte
31A5 23 15140 INC HL
31A6 23 15150 INC HL
31A7 23 15160 INC HL
15170 ;
15180 ; P/u length byte & pt to driver name
15190 ;
31A8 46 15200 LD B,(HL) ;P/u length byte
31A9 23 15210 INC HL ;HL => Driver Name
31AA 11C034 15220 LD DE,MD$ ;DE => MEMDISK
15230 ;
15240 ; Is this REALLY a certified MemDISK ??
15250 ;
31AD 1A 15260 MEMLP LD A,(DE) ;P/u MemDISK byte
31AE BE 15270 CP (HL) ;Match ?
31AF 23 15280 INC HL ;Bump driver ptr
31B0 13 15290 INC DE ;Bump string ptr
31B1 C2CE32 15300 JP NZ,NOTMEM ;No - isn't a MemDISK
31B4 10F7 15310 DJNZ MEMLP ;Yes - check all posns
15320 ;
15330 ; Pick up Old HIGH$ address & stuff for later
15340 ;
31B6 5E 15350 LD E,(HL) ;P/u old HIGH$
31B7 23 15360 INC HL
31B8 56 15370 LD D,(HL)
31B9 ED53EC31 15380 LD (SAVEOLD+1),DE ;Stuff into LD HL inst
15390 ;
15400 ; P/u BANK information
15410 ;
31BD FDCB03A6 15420 RES 4,(IY+DFLAG$) ;Reset MemDISK bit
31C1 23 15430 INC HL ;HL => Bank image
31C2 7E 15440 LD A,(HL) ;P/u bank image
31C3 4F 15450 LD C,A ;Xfer to C
31C4 FE03 15460 CP 3 ;Both banks 1 & 2 ?
31C6 3805 15470 JR C,FRBANK ;No - free up bank

```

MEMDISKA - Installation

```

31C8 0D      15480      DEC      C      ;Set C = 2
31C9 CD3C2C  15490      CALL     FREBANK ;Free bank #2
31CC 0D      15500      DEC      C      ;Set C = 1
31CD CD3C2C  15510 FRBANK CALL     FREBANK ;Free Bank in C
              15520 ;
              15530 ;      Is this a Bank Zero MemDISK ?
              15540 ;
31D0 FD215E37 15550      LD       IY,TYPEDIS ;IY => Disable Type
31D4 0C      15560      INC      C      ;Is C = 0 ?
31D5 0D      15570      DEC      C
31D6 201C    15580      JR       NZ,GTDRV2 ;No - check out driver
              15590 ;
              15600 ;      Bank 0 - p/u last HIGH$ from Driver storage
              15610 ;
31D8 FD3500  15620      DEC      (IY)      ;Change type
31DB 23      15630      INC      HL      ;Pos to HI$ val after
31DC 23      15640      INC      HL      ; MemDISK installation.
31DD 23      15650      INC      HL
31DE 5E      15660      LD       E,(HL)    ;P/u address
31DF 23      15670      INC      HL
31E0 56      15680      LD       D,(HL)
              15690 ;
              15700 ;      Pick up Current HIGH$ & compare with other
              15710 ;
31E1 60      15720      LD       H,B      ;Set HL = 0
31E2 68      15730      LD       L,B
31E3         15740      @@HIGH$      ;(B=0), p/u HIGH$
31E3 3E64    00073      LD       A,100
31E5 EF      00074      RST       40
31E6 B7      15750      OR       A      ;Same ?
31E7 ED52    15760      SBC      HL,DE
31E9 2009    15770      JR       NZ,GTDRV2 ;NZ - Can't do it
              15780 ;
              15790 ;      Reset HIGH$ = original HIGH$
              15800 ;
31EB 210000  15810 SAVEOLD LD      HL,$-$    ;P/u old HIGH$
31EE         15820      @@HIGH$      ;Re-allocate space
31EE 3E64    00075      LD       A,100
31F0 EF      00076      RST       40
31F1 FD3400  15830      INC      (IY)      ;Change Type
              15840 ;
              15850 ;      Can the Driver area be re-allocated ?
              15860 ;
31F4 CD352D  15870 GTDRV2 CALL     GTDRV    ;Get driver area
31F7 210000  15880 DREND  LD      HL,$-$    ;P/u driver address
31FA B7      15890      OR       A
31FB ED52    15900      SBC      HL,DE      ;Same ?
31FD E1      15910      POP      HL      ;HL => Driver Address
31FE 2016    15920      JR       NZ,NORECLM ;No - can't Reclaim
              15930 ;
              15940 ;      Stuff original Address into low driver ptr
              15950 ;
3200 DD7400  15960      LD       (IX),H    ;Msb
3203 DD75FF  15970      LD       (IX-1),L  ;Lsb
3206 FD3400  15980      INC      (IY)      ;Change type
3209 FD3400  15990      INC      (IY)
              16000 ;
              16010 ;      Clear out Driver
              16020 ;

```

MEMDISKA - Installation

```

320C 01DB00 16030 LD BC,LENGTH-1 ;BC = # of bytes clr
320F 3600 16040 LD (HL),0 ;Null byte
3211 54 16050 LD D,H ;Set DE = HL+1
3212 5D 16060 LD E,L
3213 13 16070 INC DE
3214 EDB0 16080 LDIR ;Clear area
16090 ;
16100 ; Disable DCT slot
16110 ;
3216 E1 16120 NORECLM POP HL ;HL => DCT + 0
3217 36C9 16130 LD (HL),0C9H ;Disable it
16140 ;
16150 ; Calculate Start of Disable string
16160 ;
3219 FDE5 16170 PUSH IY ;Xfer to HL
321B E1 16180 POP HL
321C 4E 16190 LD C,(HL) ;P/u type
321D CB21 16200 SLA C ;Multiply by 2
321F 0600 16210 LD B,0 ;BC = offset in table
3221 23 16220 INC HL ;HL => Address Table
3222 09 16230 ADD HL,BC ;HL => Add of mess string
3223 5E 16240 LD E,(HL) ;P/u Address
3224 23 16250 INC HL
3225 56 16260 LD D,(HL)
3226 EB 16270 EX DE,HL ;HL => Disable message
3227 16280 @@LOGOT ;Log message
00077 IFEQ 00H,1
00078 LD HL,
00079 ENDIF
3227 3E0C 00080 LD A,12
3229 EF 00081 RST 40
322A C3212C 16290 JP EXIT ;Go to exit routine
16300 ;
16310 ; FORMAT - Format Memory
16320 ;
322D 21B036 16330 FORMAT LD HL,VERIFY ;"Verifying RAM ..."
3230 16340 @@DSPLY ;Display it
00082 IFEQ 00H,1
00083 LD HL,
00084 ENDIF
3230 3E0A 00085 LD A,10
3232 EF 00086 RST 40
3233 1600 16350 LD D,00 ;Track counter
16360 ;
16370 ; Display Current Cylinder Formatting
16380 ;
3235 7A 16390 WIPELP LD A,D ;Get track counter
3236 CD442C 16400 CALL DECASC2 ;Display Dec ASCII equiv.
16410 ;
16420 ; Run 4 different bit tests on each cylinder
16430 ;
3239 3EFF 16440 LD A,11111111B ;All bits on
323B CD5B32 16450 CALL VERCYL ;Verify track w/ bits on
323E 3E55 16460 LD A,01010101B ;Next pattern
3240 CD5B32 16470 CALL VERCYL
3243 3EAA 16480 LD A,10101010B ;Last pattern
3245 CD5B32 16490 CALL VERCYL
3248 3E00 16500 LD A,00000000B ;All bits off
324A CD5B32 16510 CALL VERCYL ;Verify track w/ bits off

```

MEMDISKA - Installation

```

16520 ;
16530 ; Finished Formatting yet ?
16540 ;
324D 14 16550 INC D ;Bump cylinder #
324E 7A 16560 LD A,D
324F DDBE06 16570 CP (IX+6) ;Finished ?
3252 20E1 16580 JR NZ,WIPELP ;No - stop when max cyl
16590 ;
16600 ; Finished Formatting - Display message
16610 ;
3254 21CA36 16620 LD HL,FORMCOM ;"Formatting Complete"
3257 16630 @@DSPLY ;Print it
00087 IFEQ 00H,1
00088 LD HL,
00089 ENDIF
3257 3E0A 00090 LD A,10
3259 EF 00091 RST 40
325A C9 16640 RET ;Done formatting
16650 ;
16660 ; VERCYL - Verify a cylinder of RAM
16670 ;
325B 210038 16680 VERCYL LD HL,IOBUFF ;HL => I/O buffer
325E 1E00 16690 LD E,0 ;Init to sector 0
16700 ;
16710 ; Fill buffer with specified byte
16720 ;
3260 77 16730 STUFLP LD (HL),A ;Stuff into buffer
3261 2C 16740 INC L ;Bump
3262 20FC 16750 JR NZ,STUFLP ;256 bytes to fill
16760 ;
16770 ; Write the sector & read it back
16780 ;
3264 F5 16790 CYLP PUSH AF ;Save fill byte
3265 CDC12F 16800 CALL WRSEC ;Write Sector
3268 CDCA2F 16810 CALL RDSEC ;Read into other buff
326B F1 16820 POP AF ;A = Fill byte
16830 ;
16840 ; Check if sector read back has correct byte
16850 ;
326C BE 16860 CKLP CP (HL) ;Match ?
326D C28D2C 16870 JP NZ,ERROR ;No - error
3270 2C 16880 INC L ;Done with sector ?
3271 20F9 16890 JR NZ,CKLP ;256 bytes to check
16900 ;
16910 ; Advance to next sector
16920 ;
3273 7B 16930 LD A,E ;P/u sector #
3274 DDBE07 16940 CP (IX+7) ;Finished ?
3277 7E 16950 LD A,(HL) ;P/u cylinder byte
3278 13 16960 INC DE ;Bump E
3279 20E9 16970 JR NZ,CYLP ;DCT+8 sectors to check
327B C9 16980 RET ;Done - RETURN
16990 ;
17000 ; FORMTIT - Check if MemDISK has data on it
17010 ;
327C 110001 17020 FORMTIT LD DE,100H ;D = Cyl 1, Sec 0 (GAT)
327F CDCA2F 17030 CALL RDSEC ;Read BOOT sector
17040 ;
17050 ; Check GAT ID

```

MEMDISKA - Installation

```

17060 ;
3282 2ED0 17070 LD L,0D0H ;MemDISK pack name
3284 11B834 17080 LD DE,MEMDISK ;What it should be
3287 0608 17090 LD B,8 ;# of characters
17100 ;
3289 1A 17110 CKMLP LD A,(DE) ;P/u should be char
328A BE 17120 CP (HL) ;Match ?
328B 23 17130 INC HL ;Bump
328C 13 17140 INC DE
328D 200C 17150 JR NZ,NOMTCH ;No - must format
328F 10F8 17160 DJNZ CKMLP ;Yes - loop for more
17170 ;
17180 ; Already a MemDISK - Sure about formatting ?
17190 ;
3291 217434 17200 LD HL,DOFORM ;Destination ...
3294 3E01 17210 LD A,1 ;Set MemDISK in flag
3296 32EA2E 17220 LD (MEMIN1+1),A
3299 1803 17230 JR DISMES ;Display it
17240 ;
17250 ; Not a MemDISK - Do normal Prompt
17260 ;
329B 219634 17270 NOMTCH LD HL,STILLFM ;Do you wish to format ?
329E 17280 DISMES @@DSPLY ;Display message
00092 IFEQ 00H,1
00093 LD HL,
00094 ENDF
329E 3E0A 00095 LD A,10
32A0 EF 00096 RST 40
17290 ;
17300 ; Input Response
17310 ;
32A1 0601 17320 LD B,1 ;Input 1 character
32A3 E5 17330 PUSH HL ;Save message start
32A4 CDD62C 17340 CALL INPUT
32A7 7E 17350 LD A,(HL) ;P/u character
32A8 E1 17360 POP HL ;Recover message start
32A9 05 17370 DEC B ;Anything entered ?
32AA C0 17380 RET NZ ;No - RETURN NZ
17390 ;
17400 ; Set Z flag if "Y" & Reset Z if "N" entered
17410 ;
32AB CBAF 17420 RES 5,A ;Cvt to U/C
32AD FE4E 17430 CP 'N' ;<N>o ?
32AF 2805 17440 JR Z,RESZF ;RETURN NZ
32B1 FE59 17450 CP 'Y' ;<Y>es ?
32B3 C8 17460 RET Z ;RETURN Z set
32B4 18E8 17470 JR DISMES ;No - reprompt
32B6 B7 17480 RESZF OR A ;Reset Z flag
32B7 C9 17490 RET ; and RETURN
17500 ;
17510 ; Variables used
32B8 0000 17520 SAVEDE DW 0
32BA 0000 17530 SAVEDCT DW 0
32BC 0000 17540 DRADD DW 0
17550 ;
17560 ; Informative Error Display & Abort Routine
17570 ;
32BE 21D534 17580 NODRV LD HL,NODRV$
32C1 DD 17590 DB 0DDH

```

MEMDISKA - Installation

```

32C2 21F334 17600 BADDRV LD HL,BADDRV$
32C5 DD 17610 DB 0DDH
32C6 211037 17620 VIASET LD HL,VIASET$ ;Not via SYSTEM
32C9 DD 17630 DB 0DDH
32CA 219936 17640 MEMIN LD HL,MEMIN$ ;Already installed
32CD DD 17650 DB 0DDH
32CE 213235 17660 NOTMEM LD HL,NOTMEM$ ;Not a MemDISK
32D1 DD 17670 DB 0DDH
32D2 214D35 17680 NOMEM LD HL,NOMEM$ ;Insufficient Memory
32D5 DD 17690 DB 0DDH
32D6 216235 17700 NOTPRS LD HL,NOTPRS$ ;Not Present
32D9 DD 17710 DB 0DDH
32DA 216736 17720 BNKUSE LD HL,BNKUSE$ ;Bank in use
32DD DD 17730 DB 0DDH
32DE 217635 17740 NOTACT LD HL,NOTACT$ ;Cant Install
17750 ;
17760 ; Log Error Message & Abort
17770 ;
32E1 17780 @@LOGOT ;Log error message
00097 IFEQ 00H,1
00098 LD HL,
00099 ENDIF
32E1 3E0C 00100 LD A,12
32E3 EF 00101 RST 40
32E4 C31B2C 17790 JP ABORT ;Go to exit routine
17800 ;
32E7 4D 17810 HELLO$ DB 'MEMDISK'
45 4D 44 49 53 4B
32EE 17820 *GET CLIENT:3
17830 ;CLIENTS/ASM - File to establish sign-on headers
17840 ;
32EE 20 17850 DB ' - 6.2.0 - Copyright 1982/83/84 by Logical'
2D 20 36 2E 32 2E 30 20
2D 20 43 6F 70 79 72 69
67 68 74 20 31 39 38 32
2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61
6C
3318 20 17860 DB ' Systems, Inc. ',10
53 79 73 74 65 6D 73 2C
20 49 6E 63 2E 20 20 20
20 20 20 0A
17870 ;
332D 41 17880 DB 'All Rights Reserved. Licensed 1982/83/84'
6C 6C 20 52 69 67 68 74
73 20 52 65 73 65 72 76
65 64 2E 20 4C 69 63 65
6E 73 65 64 20 31 39 38
32 2F 38 33 2F 38 34
3355 20 17890 DB ' to xxxxxxxxxxxxxxxxxx',10,13
74 6F 20 78 78 78 78 78
78 78 78 78 78 78 78 78
78 78 78 78 78 0A 0D
17900 ;
336D 0A 17910 BANKS DB LF,'<A> Bank 0 (Primary Memory)',LF
3C 41 3E 20 20 42 61 6E
6B 20 30 20 28 50 72 69
6D 61 72 79 20 4D 65 6D
6F 72 79 29 0A

```

MEMDISKA - Installation

```

338B 3C      17920      DB      '<B> Bank 1',LF
      42 3E 20 20 42 61 6E 6B
      20 31 0A
3397 3C      17930      DB      '<C> Bank 2',LF
      43 3E 20 20 42 61 6E 6B
      20 32 0A
33A3 3C      17940      DB      '<D> Banks 1 and 2',LF
      44 3E 20 20 42 61 6E 6B
      73 20 31 20 61 6E 64 20
      32 0A
33B6 3C      17950      DB      '<E> Disable MemDISK',LF,LF
      45 3E 20 20 44 69 73 61
      62 6C 65 20 4D 65 6D 44
      49 53 4B 0A 0A
33CC 57      17960      DB      'Which type of allocation - '
      68 69 63 68 20 74 79 70
      65 20 6F 66 20 61 6C 6C
      6F 63 61 74 69 6F 6E 20
      2D 20
33E7 3C      17970      DB      '<A>, <B>, <C>, <D>, or <E> ? ',ETX
      41 3E 2C 20 3C 42 3E 2C
      20 3C 43 3E 2C 20 3C 44
      3E 2C 20 6F 72 20 3C 45
      3E 20 3F 20 03
      17980 ;
3405 4E      17990 FRTRACK DB      'Note: Each Cylinder equals '
      6F 74 65 3A 20 45 61 63
      68 20 43 79 6C 69 6E 64
      65 72 20 65 71 75 61 6C
      73 20
3420 34      18000 FRTRK1 DB      '4.50K of space.',LF
      2E 35 30 4B 20 6F 66 20
      73 70 61 63 65 2E 0A
3430 4E      18010      DB      'Number of free Cylinders: ',MINCYL+'0'&0FFH,'-'
      75 6D 62 65 72 20 6F 66
      20 66 72 65 65 20 43 79
      6C 69 6E 64 65 72 73 3A
      20 33 2D
344C 30      18020 FRTRK2 DB      '00 ? ',ETX
      30 20 3F 20 03
      18030 ;
3452 53      18040 DENSITY DB      'Single or Double Density <S,D> ? ',ETX
      69 6E 67 6C 65 20 6F 72
      20 44 6F 75 62 6C 65 20
      44 65 6E 73 69 74 79 20
      3C 53 2C 44 3E 20 3F 20
      03
      18050 ;
3474 44      18060 DOFORM DB      'Destination MemDISK contains Data',LF
      65 73 74 69 6E 61 74 69
      6F 6E 20 4D 65 6D 44 49
      53 4B 20 63 6F 6E 74 61
      69 6E 73 20 44 61 74 61
      0A
      18070 ;
3496 44      18080 STILLFM DB      'Do you wish to Format it <Y/N> ? ',ETX
      6F 20 79 6F 75 20 77 69
      73 68 20 74 6F 20 46 6F
      72 6D 61 74 20 69 74 20

```


MEMDISKA - Installation

```

      3C 59 2F 4E 3E 20 3F 20
      03
      18090 ;
34B8 4D      18100 MEMDISK DB      'MEMDISK '
      45 4D 44 49 53 4B 20
34C0 24      18110 MD$      DB      '$MD',ETX
      4D 44 03
34C4 18      18120 MDDATA DB      18H,17,0,0,8,'MemDISKD',0,0,0,0
      11 00 00 08 4D 65 6D 44
      49 53 4B 44 00 00 00 00
      18130 ;
34D5 4C      18140 NODRV$ DB      'Logical drive number required',CR
      6F 67 69 63 61 6C 20 64
      72 69 76 65 20 6E 75 6D
      62 65 72 20 72 65 71 75
      69 72 65 64 0D
34F3 43      18150 BADDRV$ DB      'Can''t specify SYSTEM drive slot',CR
      61 6E 27 74 20 73 70 65
      63 69 66 79 20 53 59 53
      54 45 4D 20 64 72 69 76
      65 20 73 6C 6F 74 0D
3513 4D      18160 INSTALD DB      'MemDISK Successfully Installed',CR
      65 6D 44 49 53 4B 20 53
      75 63 63 65 73 73 66 75
      6C 6C 79 20 49 6E 73 74
      61 6C 6C 65 64 0D
      18170 ;
3532 54      18180 NOTMEM$ DB      'Target Drive not a MemDISK',CR
      61 72 67 65 74 20 44 72
      69 76 65 20 6E 6F 74 20
      61 20 4D 65 6D 44 49 53
      4B 0D
      18190 ;
354D 49      18200 NOMEM$ DB      'Insufficient Memory ',CR
      6E 73 75 66 66 69 63 69
      65 6E 74 20 4D 65 6D 6F
      72 79 20 0D
      18210 ;
3562 4D      18220 NOTPRS$ DB      'MemDISK not present',CR
      65 6D 44 49 53 4B 20 6E
      6F 74 20 70 72 65 73 65
      6E 74 0D
      18230 ;
3576 4D      18240 NOTACT$ DB      'MemDISK not present, installation '
      65 6D 44 49 53 4B 20 6E
      6F 74 20 70 72 65 73 65
      6E 74 2C 20 69 6E 73 74
      61 6C 6C 61 74 69 6F 6E
      20
3598 61      18250      DB      'aborted',CR
      62 6F 72 74 65 64 0D
      18260 ;
35A0 4D      18270 DISABE1 DB      'MemDISK disabled, memory now avail'
      65 6D 44 49 53 4B 20 64
      69 73 61 62 6C 65 64 2C
      20 6D 65 6D 6F 72 79 20
      6E 6F 77 20 61 76 61 69
      6C
35C2 61      18280      DB      'able',CR

```

MEMDISKA - Installation

```

        62 6C 65 0D
                18290 ;
35C7 4D        18300 DISABE2 DB      'MemDISK disabled, Unable to reclaim '
        65 6D 44 49 53 4B 20 64
        69 73 61 62 6C 65 64 2C
        20 55 6E 61 62 6C 65 20
        74 6F 20 72 65 63 6C 61
        69 6D 20
35EB 68        18310                DB      'high memory',CR
        69 67 68 20 6D 65 6D 6F
        72 79 0D
                18320 ;
35F7 4D        18330 DISABE3 DB      'MemDISK disabled, Unable to reclaim '
        65 6D 44 49 53 4B 20 64
        69 73 61 62 6C 65 64 2C
        20 55 6E 61 62 6C 65 20
        74 6F 20 72 65 63 6C 61
        69 6D 20
361B 64        18340                DB      'driver area',CR
        72 69 76 65 72 20 61 72
        65 61 0D
                18350 ;
3627 4D        18360 DISABE4 DB      'MemDISK disabled, Unable to reclaim '
        65 6D 44 49 53 4B 20 64
        69 73 61 62 6C 65 64 2C
        20 55 6E 61 62 6C 65 20
        74 6F 20 72 65 63 6C 61
        69 6D 20
364B 68        18370                DB      'high memory and driver area',CR
        69 67 68 20 6D 65 6D 6F
        72 79 20 61 6E 64 20 64
        72 69 76 65 72 20 61 72
        65 61 0D
                18380 ;
3667 55        18390 BNKUSE$ DB      'Unable to install MemDISK, '
        6E 61 62 6C 65 20 74 6F
        20 69 6E 73 74 61 6C 6C
        20 4D 65 6D 44 49 53 4B
        2C 20
3682 72        18400                DB      'requested bank in use.',CR
        65 71 75 65 73 74 65 64
        20 62 61 6E 6B 20 69 6E
        20 75 73 65 2E 0D
                18410 ;
3699 4D        18420 MEMIN$ DB      'MemDISK already Active',CR
        65 6D 44 49 53 4B 20 61
        6C 72 65 61 64 79 20 41
        63 74 69 76 65 0D
                18430 ;
36B0 56        18440 VERIFY DB      'Verifying RAM cylinder 00',ETX
        65 72 69 66 79 69 6E 67
        20 52 41 4D 20 63 79 6C
        69 6E 64 65 72 20 30 30
        03
                18450 ;
36CA 0A        18460 FORMCOM DB      LF,'Verifying Complete, RAM good',LF
        56 65 72 69 66 79 69 6E
        67 20 43 6F 6D 70 6C 65
        74 65 2C 20 52 41 4D 20

```

MEMDISKA - Installation

```

        67 6F 6F 64 0A
36E8 44      18470 DB      'Directory has been placed on Cylinder 1',CR
        69 72 65 63 74 6F 72 79
        20 68 61 73 20 62 65 65
        6E 20 70 6C 61 63 65 64
        20 6F 6E 20 43 79 6C 69
        6E 64 65 72 20 31 0D
        18480 ;
3710 4D      18490 VIASET$ DB      'Must install via SYSTEM (DRIVER=',CR
        75 73 74 20 69 6E 73 74
        61 6C 6C 20 76 69 61 20
        53 59 53 54 45 4D 20 28
        44 52 49 56 45 52 3D 0D
        18500 ;
3731 0A      18510 BADRAM DB      LF,'Verify Error in Bank '
        56 65 72 69 66 79 20 45
        72 72 6F 72 20 69 6E 20
        42 61 6E 6B 20
3747 6E      18520 VBANK DB      'n at location X',AP
        20 61 74 20 6C 6F 63 61
        74 69 6F 6E 20 58 27
3757 6E      18530 VLOC DB      'nnnn',AP,LF,CR
        6E 6E 6E 27 0A 0D
        18540 ;
375E 01      18550 TYPEDIS DB      1 ;Type of disable
375F 2736    18560 DISTAB DW      DISABE4,DISABE3,DISABE2,DISABE1
        F735 C735 A035
3767 00      18570 RE_USE DB      0 ;Re-use trapped driver area.
        18580 ;
        18590 ;      Buffers Used
        18600 ;
3800      18610 ORG      $<-8+1<+8
        18620 ;
0100      18630 IOBUFF DS      256
0100      18640 BUFFER DS      256
000A      18650 DUPDCT DS      10
        18660 ;
        00570 ;
3A0A      00580 SUBTTL <>
2C00      00590 END      START

```

\$NOT	2F10 @@1	0000 @@2	0000
@@3	0000 @@4	0000 @MOD2	0000
@MOD4	FFFF ABB	0010 ABORT	2C1B
AP	0027 A B C	3080 B10	2E22
B13	2E29 B14	2E4B B14A	2E57
B9	2DF7 BADDRV	32C2 BADDRV\$	34F3
BADRAM	3731 BANKIM	2DC8 BANKS	336D
BNKUSE	32DA BNKUSE\$	3667 BOOT	2FFF
BOOTERN	3013 BOOTGRN	3016 BPC	2CE4
BREAK	0080 BS	0008 BTLP	2F3C
BTSECS	2F3A BUFF	2E8B BUFFER	3900
BUFFER\$	2300 CALCDRV	2CF6 CALCSIZ	3165
CALCYL	312B CFLAG\$	0002 CHKDIR	2E1A
CHKDIR2	2E19 CKBANK	2CCB CKLP	326C
CKMLP	3289 CR	000D CYLP	3264
CYLS	2F69 DDBPC	1200 DECASC	2C63
DECASC2	2C44 DECHEX	2C6E DEFAULT	30FE
DEFBANK	2E6A DENSITY	3452 DFLAG\$	0003
DIR	301F DIRERN	3033 DISABE1	35A0
DISABE2	35C7 DISABE3	35F7 DISABE4	3627
DISMEM	318A DISMES	329E DISTAB	375F
DIVLP	2CE9 DIVLP1	2D13 DOFORM	3474
DOFORM1	2F13 DOMEM	303F DONE1	2C80
DONTRES	2EFA DOXFER	2E0C DOXFER1	2D26
DO INST	2EDC DRADD	32BC DREND	31F7
DRIVE	2FC4 DRIVER	2DBE DRVLOW	2DC9
DSP	2C59 DUPDCT	3A00 ERROR	2C8D
ETX	0003 EX1	2E5B EXIT	2C21
FILBUF	2F34 FLAG	0040 FORMAT	322D
FORMCOM	36CA FORMTIT	327C FRBANK	31CD
FREBANK	2C3C FREETRK	2F8E FRTRACK	3405
FRTRK1	3420 FRTRK2	344C GAT0	2F5D
GATCD	2F71 GETADR	2E5D GETBUF	2E8A
GETCYL	2CE2 GETDIG	2C82 GETDUP	2D2C
GETOLD	2E83 GETYPE	304F GOTBANK	2E70
GPC	2F86 GTDRV	2D35 GTDRV2	31F4
HELLO\$	32E7 HIDVR	1300 ILLEGAL	2C8A
INIT	2DDD INPDENS	309B INPUT	2CD6
INSTALD	3513 INSTDRV	2D48 INSTMEM	2E9A
IOBUFF	3800 IOERR	2C5D KFLAG\$	000A
KIDCB\$	2EF7 LENGTH	00DC LF	000A
LOCKOUT	2F62 LOWEST	8000 LPADD	2C65
LSIID	2FBA MAXCYL	315E MD\$	34C0
MDDATA	34C4 MEMDISK	34B8 MEMDRV	2DF6
MEMHIGH	2DCB MEMIN	32CA MEMIN\$	3699
MEMIN1	2EE9 MEMLP	31AD MINCYL	0003
MLOOP	316B MYSTACK	2DDD NODRV	32BE
NODRV\$	34D5 NOMEM	32D2 NOMEM\$	354D
NOMTCH	329B NORECLM	3216 NORMEX	2C16
NOT2	3096 NOTACT	32DE NOTACT\$	3576
NOTDIR	2E20 NOTMEM	32CE NOTMEM\$	3532
NOTPRS	32D6 NOTPRS\$	3562 NOT_IN	2EB5
NUM	0080 OFFSET	2E65 OKTOGO	2ED6
OLDHIGH	2DC0 OLDRV	2EEE OLD_HI	2DC6
PAR_ERR	002C PIKUPHI	310F RDSEC	2FCA
RECVDE	2E3F REDO	3148 REL1	2DFF
REL2	2E0F REL2A	2E06 REL3	2E40
REL4	2E46 REL5	2E79 REL6	2DE0

REL7	2DE5 REL8	2DEB REL8A	2E35
REL8B	2E7C REL9	2DE8 RELDUN	2D8A
RELTBL	2D70 RESTREG	2CC6 RESZF	32B6
RETADDR	2CC3 RE_USE	3767 RLOOP	2D57
SAVDCT	2D20 SAVDCT	32BA SAVEDE	32B8
SAVEOLD	31EB SAVEREG	2CAE SAVESP	2DEE
SAVPAGE	316E SDBPC	0A00 SDENA	2E5E
SDENB	2E62 SDENC	2E63 SDEND	2DA2
SDENE	2DA6 SDENF	2DB1 SDENG	2DB5
SDENI	3036 SETBANK	2C28 SETDCT	2D94
SFLAG\$	0012 SHOWINU	2F22 SPC	3169
START	2C00 STARTA	2C09 STBANK	2C34
STFRET	2E72 STILLFM	3496 STR	0020
STUFLP	3260 TAB	0009 TYPED	307A
TYPEDIS	375E VBANK	3747 VERCYL	325B
VERIFY	36B0 VFLAG\$	0015 VIASET	32C6
VIASET\$	3710 VLOC	3757 WAS0	3098
WIPELP	3235 WP	000F WRBOOT	2F30
WRENT	2FE2 WRGAT	2F5A WRGAT1	2FBE
WRHIT	2FD5 WRITES	2E2D WRSEC	2FC1
ZEROHIT	2FD6 @@ABORT	7059 @@ADTSK	70EC
@@BANK	7604 @@BKSP	72E4 @@BREAK	761A
@@CHNIO	7044 @@CKBRKC	7668 @@CKDRV	7140
@@CKEOF	72F9 @@CKTSK	70D7 @@CLOSE	72CF
@@CLS	7652 @@CMNDI	7083 @@CMNDR	7098
@@CTL	6EA8 @@DATE	701A @@DCSTAT	717F
@@DEBUG	70C2 @@DECHEX	7584 @@DIRRD	74F1
@@DIRWR	7506 @@DIV16	756F @@DIV8	755A
@@DODIR	7155 @@DSP	6E6C @@DSPLY	6F0C
@@ERROR	70AD @@EXIT	706E @@FEXT	745E
@@FLAGS	75EE @@FNAME	7473 @@FSPEC	7449
@@GATRD	74DC @@GATWR	751B @@GET	6E80
@@GTDCB	749D @@GTDCT	7488 @@GTMOD	74B2
@@HDFMT	7227 @@HEX16	75C3 @@HEX8	75AE
@@HEXDEC	7599 @@HIGH\$	75D8 @@INIT	72A5
@@KBD	6EE4 @@KEY	6E58 @@KEYIN	6EF8
@@KLTSK	712B @@LOAD	741F @@LOC	730E
@@LOF	7323 @@LOGGER	6F43 @@LOGOT	6F58
@@MSG	6F8F @@MUL16	7545 @@MUL8	7530
@@OPEN	72BA @@PARAM	7005 @@PAUSE	6FF0
@@PEOF	7338 @@POSN	734D @@PRINT	6FA4
@@PRT	6EBC @@PUT	6E94 @@RAMDIR	716A
@@RDSEC	71FD @@RDSSC	74C7 @@READ	7362
@@REMOV	7290 @@RENAM	727B @@REW	7377
@@RMTSK	7101 @@RPTSK	7116 @@RREAD	738C
@@RSLCT	71E8 @@RSTOR	71A9 @@RUN	7434
@@RWRIT	73A1 @@SEEK	71D3 @@SEEKSC	73B6
@@SKIP	73CB @@SLCT	7194 @@STEPI	71BE
@@TIME	702F @@VDCTL	6FDB @@VER	73E0
@@VRSEC	7212 @@WEOF	73F5 @@WHERE	6ED0
@@WRITE	740A @@WRSEC	723C @@WRSSC	7251
@@WTRK	7266		

2C00 is the transfer address

00000 Total errors

NOTES:

NOTES:

PATCH/CMD - Disk file patch utility

Patch allows changing bytes in any type of disk file, be it a load module format file or standard data file. Patch code may be typed in on the command line or read from an ASCII disk file.


```

00100 ;PATCH/ASM
0000 00110 TITLE <PATCH - LS-DOS 6.2>
00120 ;
0003 00130 ETX EQU 3
000A 00140 LF EQU 10
000D 00150 CR EQU 13
0040 00160 FLAG EQU 01000000B
0010 00170 ABB EQU 00010000B
00180 ;
0000 00190 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00300 *LIST ON
0000 00200 *GET COPYCOM:3 ;Copyright message
00200 ; COPYCOM - File for Copyright COMMENT block
00300 ;
0000 00340 COM '<*(C) 1982,83,84 by LSI*>'
00210 ;
2600 00220 ORG 2600H
00230 ;
00240 BEGIN
2600 00250 @@CKBRKC ;Check if Break hit
2600 3E6A 00001 LD A,106
2602 EF 00002 RST 40
2603 2804 00260 JR Z,BEGINA ;Continue if no break
2605 21FFFF 00270 LD HL,-1 ; else abort
2608 C9 00280 RET
00290 ;
00300 BEGINA
2609 ED73BF27 00310 LD (STACK),SP ;Save original stack
260D E5 00320 PUSH HL ;Save ptr to CMD buffer
260E 00330 @@FLAGS ;Set up IY
260E 3E65 00003 LD A,101
2610 EF 00004 RST 40
2611 21C02D 00340 LD HL,HELLO$
2614 CD1F2D 00350 CALL $DSPLY ;Display the signon msg
00360 ;
00370 ; Get /CMD file off command line
00380 ;
2617 E1 00390 POP HL ;P/u cmd line ptr
2618 117F2D 00400 LD DE,PGMDCB ;Set up for OPEN
261B 00410 @@FSPEC ;Fetch program filespec
261B 3E4E 00005 LD A,78
261D EF 00006 RST 40
261E C2512D 00420 JP NZ,PGMREQ ;Quit if illegal name
2621 1A 00430 LD A,(DE)
2622 FE2A 00440 CP '*' ;Test for device spec
2624 CA512D 00450 JP Z,PGMREQ ;Abort if not a filespec
2627 E5 00460 PUSH HL ;Save posn on command line
2628 21792D 00470 LD HL,CMDEXT
262B 00480 @@FEXT ;Default ext to /CMD
262B 3E4F 00007 LD A,79
262D EF 00008 RST 40
262E D5 00490 PUSH DE ;Save ptr to FCB
262F EB 00500 EX DE,HL ;Pt HL at current name
2630 113330 00510 LD DE,FNM$ ;Store the name away
2633 00520 @@FSPEC ; in case of a later error
2633 3E4E 00009 LD A,78
2635 EF 00010 RST 40
2636 D1 00530 POP DE ;Recover FCB
2637 210033 00540 LD HL,PGMBUF ;Buffer for /CMD file I/O

```

The Source	UTILITY Files	PATCH - LS-DOS 6.2	Page 00002
263A 0600	00550	LD B,0	;Set lrl=256
263C CDEF2C	00560	CALL \$OPEN	;Open the file to fix
	00570 ;		
	00580 ;	Get /FIX file (if any)	
	00590 ;		
263F E1	00600	POP HL	;Get command line posn
2640 11A02D	00610	LD DE,FIXDCB	;FCB used for /FIX file
2643	00620	@@FSPEC	;See if a filespec is there
2643 3E4E	00011	LD A,78	
2645 EF	00012	RST 40	
2646 C27426	00630	JP NZ,CKLIN	;If error, ck for parms there
2649 E5	00640	PUSH HL	;Save command line posn
264A 217C2D	00650	LD HL,FIXEXT	
264D	00660	@@FEXT	;Use default ext of /FIX
264D 3E4F	00013	LD A,79	
264F EF	00014	RST 40	
2650 21A02D	00670	LD HL,FIXDCB	;Pt HL to start of fix filespec
2653 110930	00680	LD DE,NAMFIX\$;Buffer to hold filename only
2656 0600	00690	LD B,0	;Init char count to 0
	00700 ;		
	00710 ;	Save patch file name for X header	
	00720 ;		
2658 7E	00730 FXNAM	LD A,(HL)	;P/u a char of the filespec
2659 23	00740	INC HL	
265A FE2F	00750	CP '/'	;Found the /FIX ext?
265C 2811	00760	JR Z,FXNAM2	;Quit if so
265E FE3A	00770	CP ':'	;Colon yet?
2660 3808	00780	JR C,FXNAM1	;If less, must be number
2662 FE41	00790	CP 'A'	;A-Z?
2664 3809	00800	JR C,FXNAM2	;If less, done
2666 FE5B	00810	CP 'Z'+1	;If not alpha, done
2668 3005	00820	JR NC,FXNAM2	
266A 12	00830 FXNAM1	LD (DE),A	;Store the name char
266B 13	00840	INC DE	;Inc storage ptr
266C 04	00850	INC B	;Inc count of name chars
266D 18E9	00860	JR FXNAM	;Loop for more
266F 78	00870 FXNAM2	LD A,B	;Store the length of
2670 320830	00880	LD (NAMLEN\$),A	; the /FIX patch file
2673 E1	00890	POP HL	;Recover command line posn
2674 7E	00900 CKLIN	LD A,(HL)	;Test command line
2675 FE0D	00910	CP CR	; for end
2677 2845	00920	JR Z,RDFIX	;Go if found
2679 23	00930	INC HL	
267A FE20	00940	CP 20H	
267C 28F6	00950	JR Z,CKLIN	;Ignore spaces
267E FE28	00960	CP '('	;Beginning of parm?
2680 C2492D	00970	JP NZ,PRMERR	;Anything else is a parm error
	00980 ;		
	00990 ;	Test for REMOVE or special Option parameters	
	01000 ;	Ignore @@PARAM errors, as the parameters may actually	
	01010 ;	be a command line patch.	
	01020 ;		
2683 115730	01030	LD DE,PTBL\$;Parameter table
2686 E5	01040	PUSH HL	;Save command line ptr
2687 2B	01050	DEC HL	;Back up to '('
2688	01060	@@PARAM	
2688 3E11	00015	LD A,17	
268A EF	00016	RST 40	
268B E1	01070	POP HL	;Restore cmd line ptr
268C 010000	01080	LD BC,\$-\$; "Remove" parm response
268D	01090 RPARAM1	EQU \$-2	

```

268F 79      01100      LD      A,C
2690 32472C  01110      LD      (RPARM),A      ;Set Remove parm
2693 01FFFF  01120      LD      BC,-1          ;0 parm - bypass need for
2694         01130 OPARM1 EQU      $-2          ; Frr,nn line if OFF
2696 79      01140      LD      A,C
2697 32412C  01150      LD      (OPARM),A      ;Set find flag
269A CABE26  01160      JP       Z,RDFIX          ;If @PARAM was good, there is
                                ; no cmd line patch code
                                01170
                                01180 ;
                                01190 ;      Check for command line patch code (CLP)
                                01200 ;
269D 010034  01210      LD      BC,FIXDATA      ;Space allocated for /FIX data
26A0 7E      01220 CKLIN1 LD      A,(HL)      ;Get char from cmd line
26A1 FE0D    01230      CP       CR
26A3 CAB526  01240      JP       Z,CKLIN3      ;Show end of CLP
26A6 FE29    01250      CP       ')'
26A8 280B    01260      JR       Z,CKLIN3      ;End of CLP if so
26AA 23      01270      INC     HL            ;Bump buffer ptr
26AB FE3A    01280      CP       ':'          ;Separator between patches?
26AD 2002    01290      JR       NZ,CKLIN2     ;If not, store char
26AF 3E0D    01300      LD      A,CR          ; else show end of this CLP
26B1 02      01310 CKLIN2 LD      (BC),A      ;Put byte into fix data buff
26B2 03      01320      INC     BC            ;Bump buff ptr
26B3 18EB    01330      JR       CKLIN1        ;Loop til end of cmd line
                                01340 ;
26B5 3E0D    01350 CKLIN3 LD      A,CR          ;Put CR into
26B7 02      01360      LD      (BC),A        ; CLP buffer
26B8 03      01370      INC     BC
26B9 3E03    01380      LD      A,ETX         ;End buffer with ETX
26BB 02      01390      LD      (BC),A
26BC 1839    01400      JR       DOFIX         ;Start patching...
                                01410 ;
                                01420 ;      P/u the fix info from the FIX file, rather than
                                01430 ;      the command line.
                                01440 ;
26BE 3A0830  01450 RDFIX LD      A,(NAMLEN$)   ;P/u len of /FIX filename
26C1 B7      01460      OR       A
26C2 CA512D  01470      JP       Z,PGMREQ      ;If none used, abort
26C5 FDCB12C6 01480      SET     0,(IY+'S'-'A') ;Set open inhibit bit
26C9 11A02D  01490      LD      DE,FXDCB      ;Set up & open /FIX file
26CC 210031  01500      LD      HL,FXBUF
26CF 0600    01510      LD      B,0
26D1 CDEF2C  01520      CALL    $OPEN
26D4 210048  01530      LD      HL,PGMDATA     ;Pt HL to highest byte avail
26D7 2B      01540      DEC     HL            ; for fix data
26D8 010034  01550      LD      BC,FIXDATA     ;Start of /FIX data storage
26DB CD072D  01560 RDFIX1 CALL    $GET1      ;Get a char fm /FIX file
26DE 200F    01570      JR       NZ,RDFIX2     ;Jump on error
26E0 E67F    01580      AND     7FH          ;Strip bit 7
26E2 2810    01590      JR       Z,RDFIX3     ;Take 0 as EOF also
26E4 02      01600      LD      (BC),A        ;Save fix data char
26E5 03      01610      INC     BC            ;Advance buffer
26E6 E5      01620      PUSH    HL            ;Save HL tempy
26E7 ED42    01630      SBC     HL,BC         ;Room in fixdata buffer?
26E9 E1      01640      POP     HL
26EA DA4D2D  01650      JP       C,TOOBIG      ;Abort if patch data too large
26ED 18EC    01660      JR       RDFIX1        ; else loop til EOF
                                01670 ;
26EF FE1C    01680 RDFIX2 CP       1CH         ;End of file?
26F1 C2322D  01690      JP       NZ,IOERR      ;Abort if not
26F4 3E03    01700 RDFIX3 LD      A,ETX         ;Mark the end of the fix data

```

```

26F6 02      01710      LD      (BC),A
              01720 ;
              01730 ;      Start patching the target file
              01740 ;

26F7 210034  01750 DOFIX LD      HL, FIXDATA      ;Pt to start of fix data
              01760 ;

26FA E5      01770 DOFIX1 PUSH   HL
26FB 21882E  01780      LD      HL, RDGINP$      ;"reading input..."
26FE CD1F2D  01790      CALL   $DSPLY
2701 E1      01800      POP     HL
2702 226B2D  01810      LD      (SETMSG+1),HL      ;Used if error in line
2705 3E00    01820      LD      A,$-$
2706         01830 PASS2 EQU     $-1      ;Zero if 1st pass thru data
2707 B7      01840      OR      A
2708 7E      01850      LD      A,(HL)      ;P/U a character
2709 CA142C  01860      JP      Z,PASS1      ;Go if 1st pass
270C 7E      01870      LD      A,(HL)
270D FE03    01880      CP      ETX      ;End of patch?
270F 285C    01890      JR      Z,PCHDUN
2711 FE2E    01900      CP      '.'      ;Comment?
2713 CAC527  01910      JP      Z,COMMENT
2716 CBAF    01920      RES     5,A      ;Make upper case
2718 FE46    01930      CP      'F'      ;FIND line?
271A CAC527  01940      JP      Z,COMMENT      ;Skip on 2nd pass or if 0=N
271D FE44    01950      CP      'D'      ;Start of D line?
271F CAD727  01960      JP      Z,DVERB
2722 FE59    01970      CP      'Y'      ;Yank previous patch?
2724 CA4228  01980      JP      Z,YANK
2727 FE4C    01990      CP      'L'      ;Library overlay?
2729 CA0029  02000      JP      Z,LVERB
272C FE52    02010      CP      'R'      ;Remove parm ?
272E CA3A28  02020      JP      Z,REMOVE
2731 FE4F    02030      CP      'O'      ;O parm ?
2733 CAD128  02040      JP      Z,OVERB
2736 FE58    02050      CP      'X'      ;Start of X line?
2738 C2612D  02060      JP      NZ,PCHERR      ;Error if none of above
              02070 ;
              02080 ;      Verb = 'X' -> patch by hex load address
              02090 ;

273B 117F2D  02100      LD      DE,PGMDCB      ;Rewind the program to 0
273E 010000  02110      LD      BC,0      ;Use POSN so EOF
2741 CDF52C  02120      CALL   $POSN      ; is not changed
2744 CD4D29  02130      CALL   POSFIL      ;Posn to end of prgfile
              02140 ;

2747 F5      02150      PUSH   AF      ;Save regs fm display routine
2748 E5      02160      PUSH   HL
2749 D5      02170      PUSH   DE
274A 21AD2E  02180      LD      HL,INSPCH$      ;"installing patch..."
274D CD1F2D  02190      CALL   $DSPLY
2750 D1      02200      POP     DE
2751 E1      02210      POP     HL
2752 F1      02220      POP     AF
              02230 ;

2753 FE02    02240      CP      2      ;Be sure type byte = 2
2755 C2452D  02250      JP      NZ,FILERR      ;Load file format error
2758 3E01    02260      LD      A,1      ;Tempy set LRL to 1
275A 32882D  02270      LD      (PGMDCB+9),A      ; & backspace the file
275D CDFB2C  02280      CALL   $BKSP      ; to overwrite old xfer addr
2760 AF      02290      XOR     A      ;Reset LRL to 256
2761 32882D  02300      LD      (PGMDCB+9),A
              02310 ;

```

```

02320 ;      Install the X patch at the end of the prgfile
02330 ;
2764 CD8229 02340 CALL STUFNM      ;Generate the patch
2767 7E      02350 LD A,(HL)      ;HL => ending posn in fix data
2768 FE03    02360 CP ETX          ;Did it go til the end?
276A C2612D 02370 JP NZ,PCHERR    ;"Patch format error..."
02380 ;
02390 ;      Patch/operation complete - close the file
02400 ;
276D 3E0D    02410 PCHDUN LD A,CR      ;Move cursor to next line
276F CD2B2D 02420 CALL $DSP
2772 117F2D 02430 LD DE,PGMDCB    ;Close the program file
2775        02440 @@CLOSE
2775 3E3C     00017 LD A,60
2777 EF      00018 RST 40
2778 C2322D 02450 JP NZ,IOERR
277B 21EE2F 02460 LD HL,YANKMSG    ;Set up in case Yank was done
277E 3A732D 02470 LD A,(YNKFLG) ;Was it a Yank?
2781 B7      02480 OR A
2782 2024    02490 JR NZ,EXLOG    ;Yes, log out
2784 21962F 02500 LD HL,SUCCESS ;"function completed.."
2787        02510 @@LOGOT
00019 IFEQ 00H,1
00020 LD HL,
00021 ENDIF
2787 3E0C     00022 LD A,12
2789 EF      00023 RST 40
278A 2A742D 02520 LD HL,(LINCNT) ;P/u # of D & X lines
278D 7C      02530 LD A,H
278E B5      02540 OR L          ;Any?
278F 2814    02550 JR Z,NOCHG    ;No D or X verbs
2791 E5      02560 PUSH HL        ;Save line count
2792 110100 02570 LD DE,1        ;Exactly 1 line?
2795 ED52    02580 SBC HL,DE
2797 E1      02590 POP HL
2798 2005    02600 JR NZ,NTONE    ;Go if more than 1
279A 3E20    02610 LD A,' '      ; else remove "s" from message
279C 32C12F 02620 LD (PLURAL),A
279F 11B12F 02630 LD DE,LINMSG$ ;Put line count into message
27A2        02640 @@HEXDEC    ; as decimal ASCII
27A2 3E61    00024 LD A,97
27A4 EF      00025 RST 40
27A5 21B12F 02650 LD HL,LINMSG$
27A8        02660 EXLOG @@LOGOT ;Show how many lines done
00026 IFEQ 00H,1
00027 LD HL,
00028 ENDIF
27A8 3E0C     00029 LD A,12
27AA EF      00030 RST 40
02670 ;
27AB 210000 02680 LD HL,0        ;Init no error
27AE E5      02690 $QUIT PUSH HL
27AF 217F2D 02700 LD HL,PGMDCB
27B2 CB7E    02710 BIT 7,(HL)    ;Was file left open?
27B4 EB      02720 EX DE,HL    ;DE=>DCB possible close
27B5 C4D62C 02730 CALL NZ,FLOPN ;Warn user
27B8 3E0E    02740 LD A,14    ;Cursor on
27BA CD2B2D 02750 CALL $DSP
27BD E1      02760 POP HL
27BE 310000 02770 LD SP,$-$    ;P/u original stack
27BF        02780 STACK EQU $-2

```

```

27C1      02790      @@CKBRKC      ;Clear break
27C1 3E6A      00031      LD      A,106
27C3 EF      00032      RST      40
27C4 C9      02800      RET      ;Done with the patching
      02810 ;
      02820 ;      Verb = '.' => comment line
      02830 ;      HL = start of line in fix data
      02840 ;      Bypass all chars until a terminator is found
      02850 ;
27C5 7E      02860 COMMENT LD      A,(HL)      ;Look for some terminator
27C6 FE03      02870      CP      ETX      ;End of the fix data?
27C8 CAFE26      02880      JP      Z,DOFIX1      ;Back if so
27CB 23      02890      INC     HL      ; else bump buffer ptr
27CC FE3B      02900      CP      ','      ;Logical EOL?
27CE 2804      02910      JR      Z,EOL1      ;Back if so
27D0 FE0D      02920      CP      CR      ;Physical EOL?
27D2 20F1      02930      JR      NZ,COMMENT      ;Do next char if not
27D4 C3FA26      02940 EOL1      JP      DOFIX1      ;Back to the caller
      02950 ;
      02960 ;      Verb = 'D' -> disk record patch
      02970 ;
27D7 CDCC2C      02980 DVERB      CALL     CNTLIN      ;Bump line counter
27DA CDE327      02990      CALL     DPOSN      ;Posn prgfile to Drr,bb
27DD CD0A28      03000      CALL     DLINE      ;Put or check the patch line
      03010 ;      ; depending on which pass
27E0 C3FA26      03020      JP      DOFIX1      ;Do next line
      03030 ;
27E3 23      03040 DPOSN      INC     HL      ;Bump fix data buffer ptr
27E4 CD922A      03050      CALL     PRSFIX      ;Get char or hex pair
27E7 0600      03060      LD      B,0      ;Put disk record #
27E9 4F      03070      LD      C,A      ; into BC
27EA 7E      03080      LD      A,(HL)      ;If no comma, then
27EB FE2C      03090      CP      ','      ; get 3rd & 4th digits
27ED 2804      03100      JR      Z,DVERB1      ; in case user put in
27EF CD922A      03110      CALL     PRSFIX      ; a 4 byte record #
27F2 4F      03120      LD      C,A
27F3 117F2D      03130 DVERB1      LD      DE,PGMDCB      ;Position file to record
27F6 CDF52C      03140      CALL     $POSN
27F9 7E      03150      LD      A,(HL)      ;Check for ',' separator
27FA FE2C      03160      CP      ','      ; between record and offset
27FC C2612D      03170      JP      NZ,PCHERR      ;Abort if not found
27FF 23      03180      INC     HL      ;Pt to offset bytes
2800 CD252D      03190      CALL     $READ      ;Read the sector
2803 CD922A      03200      CALL     PRSFIX      ;Make offset binary in A
2806 32842D      03210      LD      (PGMDCB+5),A      ;Set byte offset in FCB
2809 C9      03220      RET
      03230 ;
      03240 ;      Dual purpose routine that checks a Drr,bb line
      03250 ;      or installs it into the program file
      03260 ;
280A 7E      03270 DLINE      LD      A,(HL)      ;Next byte in line must
280B FE3D      03280      CP      '='      ; be '='
280D C2612D      03290      JP      NZ,PCHERR      ;Abort if missing
2810 23      03300 DVERB2      INC     HL      ;Pt to start of patch data
2811 CD962A      03310 DVERB3      CALL     PRSFX1      ;Get patch byte as binary in A
2814 CDB82C      03320      CALL     PUTORCHK      ;Either write it or check it
2817 7E      03330      LD      A,(HL)      ;P/u next char
2818 FE0D      03340      CP      CR      ;Go on CR
281A 2811      03350      JR      Z,DVERB4A
281C FE3B      03360      CP      ','      ;End of logical line?
281E 280C      03370      JR      Z,DVERB4

```

```

2820 FE22      03380      CP      '""      ;Closing dbl-quote?
2822 2808      03390      JR      Z,DVERB4
2824 3AAB2A    03400      LD      A,(STRFLG+1)      ;If in quote string,
2827 B7        03410      OR      A      ; do not bump HL past
2828 28E6      03420      JR      Z,DVERB2      ; the non-existant space
282A 18E5      03430      JR      DVERB3      ;No special, do next byte
                03440 ;
282C 7E        03450 DVERB4 LD      A,(HL)      ;Ignore rest of line
282D 23        03460 DVERB4A INC     HL
282E FE0D      03470      CP      CR
2830 20FA      03480      JR      NZ,DVERB4      ;Loop til physical EOL
2832 3A0627    03490      LD      A,(PASS2)      ;Patching or checking?
2835 B7        03500      OR      A      ;If patching, need to
2836 C4012D    03510      CALL    NZ,$RWIT      ; re-write the sector
2839 C9        03520      RET      ;Done w/line
                03530 ;
                03540 ;      Verb = 'R' -> set flag to yank D patch
                03550 ;      This routine is needed to check the R parm
                03560 ;      when doing a CLP, in case the parm was entered
                03570 ;      after the fix data
                03580 ;
283A 3EFF      03590 REMOVE LD      A,-1      ;Set Reomve parm true and
283C 32472C    03600      LD      (RPARM),A      ; then ignore all until the
283F C3C527    03610      JP      COMMENT      ; next logical line
                03620 ;
                03630 ;      Verb = 'Y' -> yanks patch with same name
                03640 ;
2842 7E        03650 YANK  LD      A,(HL)      ;Ignore all chars until
2843 23        03660      INC     HL      ; the physical EOL
2844 FE0D      03670      CP      CR
2846 20FA      03680      JR      NZ,YANK
                03690 ;
2848 E5        03700      PUSH    HL      ;Save fix data posn
2849 21DC2E    03710      LD      HL,YNKPCH$      ;"yanking patch..."
284C CD1F2D    03720      CALL    $DSPLY
284F 010000    03730      LD      BC,0      ;Rewind the file
2852 117F2D    03740      LD      DE,PGMDCB
2855 CDF52C    03750      CALL    $POSN
2858 CD072D    03760 YANK1 CALL    $GET1      ;Get a "type" byte
285B C2C628    03770      JP      NZ,YANK9      ;If error, ck for EOF
285E FE07      03780      CP      7      ;Found a patch?
2860 281B      03790      JR      Z,YANK4      ;If so, check name
2862 326A28    03800      LD      (TYPCOD+1),A      ;Stuff type for testing
2865 CD0B2D    03810      CALL    $GET      ;Get a block length
2868 47        03820      LD      B,A      ;Set loop counter
2869 3E00      03830 TYPCOD LD      A,0      ;Test type
286B 3D        03840      DEC     A      ;Ck for type 1 (code block)
286C 2008      03850      JR      NZ,YANK2      ;Length ok if not
                03860 ;
                03870 ;      Adjust length for 255 & 256 byte code blocks
                03880 ;
286E CD0B2D    03890      CALL    $GET      ;Read 1st two bytes
2871 05        03900      DEC     B      ; in case the block was
2872 CD0B2D    03910      CALL    $GET      ; 255+2 or 256+2
2875 05        03920      DEC     B
2876 CD0B2D    03930 YANK2 CALL    $GET      ;Read rest of block
2879 10FB      03940 YANK3 DJNZ    YANK2
287B 18DB      03950      JR      YANK1
                03960 ;
                03970 ;      Found patch code area, is this the one?
                03980 ;

```

The Source	UTILITY Files	PATCH - LS-DOS 6.2	Page 00008
287D CD0B2D	03990 YANK4	CALL \$GET	;Get name len fm file
2880 47	04000	LD B,A	;Save len in B
2881 3A0830	04010	LD A,(NAMLEN\$)	;P/u fix file name length
2884 B8	04020	CP B	;If no match, not fix
2885 20EF	04030	JR NZ,YANK2	; to Yank
2887 210930	04040	LD HL,NAMFIX\$;Pt to yank file name
288A CD072D	04050 YANK5	CALL \$GET1	;Ck for match of yank
288D C27628	04060	JP NZ,YANK2	; file name with prog
2890 BE	04070	CP (HL)	; patch name
2891 23	04080	INC HL	
2892 20E5	04090	JR NZ,YANK3	;Back if no match
2894 10F4	04100	DJNZ YANK5	
	04110 ;		
	04120 ;	Found this fix patch - let's yank it	
	04130 ;		
2896 CD0B2D	04140 YANK6	CALL \$GET	;Get type code
2899 FE01	04150	CP 1	;Ignore block if
289B C2BD28	04160	JP NZ,YANK8	; type <> 1 (code block)
289E 3E01	04170	LD A,1	;Set LRL=1 & backspace
28A0 32882D	04180	LD (PGMDCB+9),A	; to overwrite the type byte
28A3 CDFB2C	04190	CALL \$BKSP	
28A6 AF	04200	XOR A	;Set LRL back to 256
28A7 32882D	04210	LD (PGMDCB+9),A	
28AA 3E10	04220	LD A,10H	;Change type=1 to =16
28AC CD112D	04230	CALL \$PUT	; and write to prgfile
28AF CD012D	04240	CALL \$RWRIT	;Force re-write
28B2 CD0B2D	04250	CALL \$GET	;Get length byte
28B5 47	04260	LD B,A	; of patch code block
28B6 CD0B2D	04270 YANK7	CALL \$GET	
28B9 10FB	04280	DJNZ YANK7	;Posn past the code block
28BB 18D9	04290	JR YANK6	;Loop through patch blocks
	04300 ;		
28BD E1	04310 YANK8	POP HL	;Not type 1, done with yank
28BE 3EFF	04320	LD A,0FFH	;Set Yank flag for
28C0 32732D	04330	LD (YNKFLG),A	; exit message dsply
28C3 C36D27	04340	JP PCHDUN	
	04350 ;		
28C6 FE1C	04360 YANK9	CP 1CH	;Got \$GET error, was EOF?
28C8 C2322D	04370	JP NZ,IOERR	;Abort if not, else
28CB 21F72E	04380	LD HL,NOYANK\$; "can't yank, not in file
28CE C3542D	04390	JP ERREXIT	
	04400 ;		
	04410 ;	Verb = '0' -> turn FIND on/off	
	04420 ;	Check special 0 parameter, determine ON or OFF	
	04430 ;		
28D1 23	04440 OVERB	INC HL	;Move past 0
28D2 7E	04450	LD A,(HL)	
28D3 FE3D	04460	CP '='	;Next char must be '='
28D5 201D	04470	JR NZ,WHATIS	; or is an error
28D7 23	04480	INC HL	;Bypass the '='
28D8 7E	04490	LD A,(HL)	
28D9 FE0D	04500	CP CR	;Was it CR or ')'?
28DB 281B	04510	JR Z,OISOFF	;0=<enter> is OFF
28DD CBAF	04520	RES 5,A	;Make Upper case
28DF FE4E	04530	CP 'N'	
28E1 2815	04540	JR Z,OISOFF	;0=N,NO etc.
28E3 FE59	04550	CP 'Y'	;Y=yes
28E5 2810	04560	JR Z,OISON	
28E7 FE4F	04570	CP 'O'	
28E9 2009	04580	JR NZ,WHATIS	;Not Y/N/ON/OFF!
28EB CDB32C	04590	CALL GETNXT	;Get nxt, already UC

The Source	UTILITY Files	PATCH - LS-DOS 6.2	Page 00009
28EE FE46	04600	CP 'F'	
28F0 2806	04610	JR Z,OISOFF	;OFF
28F2 FE4E	04620	CP 'N'	
28F4 C2612D	04630	JP NZ,PCHERR	;Quit if no acceptable flag
	04640 ;		
28F7 3E	04650	OISON DB 3EH	;LD A,0AFH
28F8 AF	04660	OISOFF XOR A	
28F9 32412C	04670	LD (OPARM),A	;Set parm on or off
28FC 2B	04680	DEC HL	
28FD C3C527	04690	JP COMMENT	;Ignore rest til logical EOL
	04700 ;		
	04710 ;	Verb = 'L' -> indicate patch to library file	
	04720 ;		
2900 23	04730	LVERB INC HL	;Bypass the 'L'
2901 CD922A	04740	CALL PRSFIX	;Get a hex digit pair
2904 4F	04750	LD C,A	;Stuff for later
2905 328F2B	04760	LD (OVRLY+1),A	
2908 7E	04770	LD A,(HL)	;Ck for end of line
2909 23	04780	INC HL	
290A FE0D	04790	CP CR	
290C C2612D	04800	JP NZ,PCHERR	;Error if not
290F CDE92A	04810	CALL FISAM	;Get isam overlay ptrs
2912 F5	04820	PUSH AF	;Save byte offset
2913 3A802D	04830	LD A,(PGMDCB+1)	
2916 C8BF	04840	RES 7,A	;Sector operations only
2918 32802D	04850	LD (PGMDCB+1),A	
291B 117F2D	04860	LD DE,PGMDCB	;Position the file to
291E CDF52C	04870	CALL \$POSN	;Overlay requested
2921 CD252D	04880	CALL \$READ	;Read in the sector
2924 F1	04890	POP AF	
2925 32842D	04900	LD (PGMDCB+5),A	;Stuff byte offset in FCB
2928 CD4D29	04910	CALL POSFIL	;Adv "positioning..."
292B FE04	04920	CP 4	;End of ISAM overlay?
292D C2452D	04930	JP NZ,FILERR	;If not, "load format er."
2930 3E01	04940	LD A,1	;Set LRL=1
2932 32882D	04950	LD (PGMDCB+9),A	
2935 CDFB2C	04960	CALL \$BKSP	;Backspace over the 4
2938 AF	04970	XOR A	;Now set LRL back to 256
2939 32882D	04980	LD (PGMDCB+9),A	
293C CD8229	04990	CALL STUFNM	;Do the patch
293F E5	05000	PUSH HL	
2940 21C12E	05010	LD HL,BLDMAP\$; "rebuilding library map."
2943 CD1F2D	05020	CALL \$DSPLY	
2946 CD3A2B	05030	CALL RPRMAP	;Rebuild the map
2949 E1	05040	POP HL	
294A C3FA26	05050	JP DOFIX1	;Loop
	05060 ;		
	05070 ;	Include the rest of Patch/Command	
	05080 ;		
294D	05090	*GET PATCHA:3	
	03950	;PATCHA/ASM - Continuation of Patch Program	
	03960 ;		
	03970 ;	Routine to position to file's end	
	03980 ;		
294D E5	03990	POSFIL PUSH HL	;Save fm display call
294E D5	04000	PUSH DE	
294F 216F2E	04010	LD HL,POSLED\$; "positioning ..."
2952 CD1F2D	04020	CALL \$DSPLY	
2955 D1	04030	POP DE	
2956 E1	04040	POP HL	
	04050 ;		

The Source	UTILITY Files	PATCH - LS-DOS 6.2	Page 00010
2957 CD0B2D	04060 POSFIL1	CALL \$GET	;Get a type byte
295A FE20	04070	CP 20H	;X'20' & up are illegal
295C D2452D	04080	JP NC,FILERR	
295F FE02	04090	CP 2	;Transfer address?
2961 C8	04100	RET Z	
2962 FE03	04110	CP 3	;Not really used in
2964 C8	04120	RET Z	; a file, yet...
2965 FE04	04130	CP 4	;End of ISAM member?
2967 C8	04140	RET Z	
2968 FE0A	04150	CP 0AH	;End of ISAM directory?
296A C8	04160	RET Z	
296B 4F	04170	LD C,A	;Save type byte
296C CD0B2D	04180	CALL \$GET	;Get block length
296F 47	04190	LD B,A	;Save it for countdown
2970 0D	04200	DEC C	;Was type = 1 ?
2971 2008	04210	JR NZ,POSFIL2	;Jump if not
2973 CD0B2D	04220	CALL \$GET	;Read off the load addr
2976 05	04230	DEC B	;Adjust length for each
2977 CD0B2D	04240	CALL \$GET	
297A 05	04250	DEC B	
297B CD0B2D	04260 POSFIL2	CALL \$GET	;Read the block
297E 10FB	04270	DJNZ POSFIL2	
2980 18D5	04280	JR POSFIL1	;Loop to next type code
	04290 ;		
	04300 ;		
	04310 ;		
	04320 ;		
			Routine to put the patch name header block into the prg data buffer and then position to the next X'' line
2982 E5	04330 STUFNM	PUSH HL	;Save posn in fix data
2983 21992E	04340	LD HL,GENPCH\$; "generating patch...
2986 CD1F2D	04350	CALL \$DSPLY	
2989 110048	04360	LD DE,PGMDATA	
298C 210830	04370	LD HL,NAMLEN\$;Pt to fix name field
298F 7E	04380	LD A,(HL)	
2990 B7	04390	OR A	
2991 280C	04400	JR Z,STUFNM2	;Go if no name len
2993 3E07	04410	LD A,7	;Set fix patch type
2995 12	04420	LD (DE),A	
2996 13	04430	INC DE	
2997 46	04440	LD B,(HL)	;Set header length
2998 04	04450	INC B	;Bump to write length
2999 7E	04460 STUFNM1	LD A,(HL)	;P/u name byte
299A 23	04470	INC HL	
299B 12	04480	LD (DE),A	;Put in output buffer
299C 13	04490	INC DE	
299D 10FA	04500	DJNZ STUFNM1	;Loop for namelen
	04510 ;		
299F E1	04520 STUFNM2	POP HL	;Recover posn in fix data
29A0 226B2D	04530 STUFNM3	LD (SETMSG+1),HL	;Start of this line
29A3 7E	04540	LD A,(HL)	
29A4 FE03	04550	CP ETX	;End of fix data?
29A6 CA2F2A	04560	JP Z,RIPPLE	
29A9 23	04570	INC HL	
29AA FE2E	04580	CP '.'	;Comment?
29AC 2809	04590	JR Z,STUFNM4	
29AE CBAF	04600	RES 5,A	;In case lower case
29B0 FE58	04610	CP 'X'	;Start of code line?
29B2 2810	04620	JR Z,DOXVB	;Go if so
29B4 C3612D	04630	JP PCHERR	; "patch input format err
	04640 ;		
29B7 7E	04650 STUFNM4	LD A,(HL)	;In a comment, loop until
29B8 23	04660	INC HL	; end of line

```

29B9 FE03      04670 CP      ETX      ;End of patch code?
29BB CA612D    04680 JP      Z,PCHERR ;Abort if so
29BE FE0D      04690 CP      CR       ;EOL?
29C0 20F5      04700 JR      NZ,STUFNM4 ;Loop if not
29C2 18DC      04710 JR      STUFNM3
                04720 ;
                04730 ;      Do the 'X' verb patch
                04740 ;      HL => Fix data buffer
                04750 ;      DE => Program data buffer
                04760 ;
29C4 CDCC2C    04770 DOXVB CALL    CNTLIN    ;Count installed lines
29C7 3E01      04780 LD      A,1      ;Show type 1 (code block)
29C9 12        04790 LD      (DE),A    ;Put in output buffer
29CA 13        04800 INC      DE
29CB D5        04810 PUSH    DE      ;Save ptr to length
29CC 13        04820 INC      DE
29CD 7E        04830 LD      A,(HL)    ;Should be ""
29CE 23        04840 INC      HL      ; around address (X'nnnn')
29CF FE27      04850 CP      27H
29D1 C2612D    04860 JP      NZ,PCHERR ;Error if not
29D4 CD922A    04870 CALL    PRSFX    ;P/u hex digit pair
29D7 47        04880 LD      B,A      ;Save hi-order address
29D8 CD922A    04890 CALL    PRSFX    ;P/u hex digit pair
29DB 12        04900 LD      (DE),A    ;Stuff lo-order address
29DC 13        04910 INC      DE
29DD 78        04920 LD      A,B
29DE 12        04930 LD      (DE),A    ;Stuff hi-order address
29DF 13        04940 INC      DE
29E0 7E        04950 LD      A,(HL)    ;Syntax requires "=" or
29E1 FE3D      04960 CP      '='      ; "" next
29E3 2806      04970 JR      Z,DOXVB1
29E5 FE27      04980 CP      27H      ;Bypass optional clng '
29E7 C2612D    04990 JP      NZ,PCHERR ;Error if not ',='
29EA 23        05000 INC      HL
29EB 23        05010 DOXVB1 INC     HL      ;Bypass the '='
29EC 0602      05020 LD      B,2      ;Len of bytes already stuffed
29EE 7E        05030 DOXVB2 LD      A,(HL)    ;Get char of fix data
29EF FE22      05040 CP      '"'      ;ASCII string?
29F1 281F      05050 JR      Z,DOXVB5    ;Go process if so
29F3 7E        05060 DOXVB3 LD      A,(HL)    ;P/u line byte
29F4 23        05070 INC      HL
29F5 FE3B      05080 CP      ';'      ;Logical end?
29F7 2811      05090 JR      Z,DOXVB4    ;Ignore trailing
29F9 FE0D      05100 CP      CR       ;End of line?
29FB 282C      05110 JR      Z,DOXVB6
29FD FE20      05120 CP      20H
29FF 28ED      05130 JR      Z,DOXVB2    ;Ignore spaces
2A01 2B        05140 DEC      HL      ;Back up, its a byte
2A02 CD962A    05150 CALL    PRSFX1    ;Get the hex digit pair
2A05 12        05160 LD      (DE),A    ;Stuff into code buffer
2A06 13        05170 INC      DE
2A07 04        05180 INC      B      ;Bump block length
2A08 18E9      05190 JR      DOXVB3
                05200 ;
                05210 ;      Bypass until end of line
                05220 ;
2A0A 7E        05230 DOXVB4 LD      A,(HL)    ;P/u the character
2A0B 23        05240 INC      HL
2A0C FE0D      05250 CP      CR       ;End of line?
2A0E 20FA      05260 JR      NZ,DOXVB4
2A10 1817      05270 JR      DOXVB6

```

```

05280 ;
05290 ;           Fix has double quote string
05300 ;
2A12 23      05310 DOXVB5 INC     HL
2A13 7E      05320          LD     A,(HL)           ;Get next char
2A14 FE03    05330          CP     ETX             ;End of fix data?
2A16 CA612D  05340          JP     Z,PCHERR         ;Can't end w/o some EOL
2A19 23      05350          INC     HL
2A1A FE0D    05360          CP     CR              ;End of line?
2A1C CA292A  05370          JP     Z,DOXVB6         ;Valid end
2A1F FE22    05380          CP     '"'            ;Closing quote?
2A21 28D0    05390          JR     Z,DOXVB3         ;Go for more
2A23 2B      05400          DEC     HL
2A24 12      05410          LD     (DE),A          ;Stuff the char
2A25 13      05420          INC     DE
2A26 04      05430          INC     B              ;Bump counter
2A27 18E9    05440          JR     DOXVB5           ;Loop until end or "
05450 ;
05460 ;           Found valid end - update length
05470 ;
2A29 E3      05480 DOXVB6 EX     (SP),HL          ;Grab length pointer
2A2A 70      05490          LD     (HL),B          ;Stuff the length
2A2B E1      05500          POP     HL
2A2C C3A029  05510          JP     STUFNM3          ;Go for more lines
05520 ;
05530 ;           Got to the end of the fix input
05540 ;
2A2F E5      05550 RIPPLE PUSH    HL
2A30 EB      05560          EX     DE,HL           ;Last patch byte to HL
2A31 110048  05570          LD     DE,PGMDATA       ;Pt to patch code buffer
2A34 AF      05580          XOR     A
2A35 ED52    05590          SBC     HL,DE           ;Calc length of fixup
2A37 22C62B  05600          LD     (RPRMAP9+1),HL        ;Stuff for later
2A3A 21AD2E  05610          LD     HL,INSPCH$          ;"installing patch
2A3D CD1F2D  05620          CALL  $DSPLY
2A40 217F2D  05630          LD     HL,PGMDCB          ;Move prog into fix
2A43 11A02D  05640          LD     DE,FIXDCB          ; file control block
2A46 012000  05650          LD     BC,32           ; for output use
2A49 EDB0    05660          LDIR
2A4B 210032  05670          LD     HL,LIBBUF          ;Set the i/o buffer
2A4E 22A32D  05680          LD     (FIXDCB+3),HL
2A51 11A02D  05690          LD     DE,FIXDCB          ;Reread the last program
2A54         05700          @RREAD          ; sector
2A54 3E45    05733          LD     A,69
2A56 EF      05734          RST     40
2A57 C2322D  05710          JP     NZ,IOERR          ;Quit on read error
05720 ;
05730 ;           Now ripple the file down while stuffing bytes
05740 ;
2A5A 210048  05750          LD     HL,PGMDATA       ;Beginning of "fixed" code
2A5D 11A02D  05760 RIPPL1 LD     DE,FIXDCB          ;Get prog byte
2A60 CD072D  05770          CALL  $GET1
2A63 C27A2A  05780          JP     NZ,RIPPL2
2A66 E5      05790          PUSH    HL          ;Save buffer ptr & byte
2A67 F5      05800          PUSH    AF
2A68 117F2D  05810          LD     DE,PGMDCB          ;Use the output fcb
2A6B 7E      05820          LD     A,(HL)           ;P/u byte from fixbuf
2A6C CD112D  05830          CALL  $PUT           ;Put to disk
2A6F ED4BC62B 05840          LD     BC,(RPRMAP9+1) ;Pt to patch length
2A73 09      05850          ADD     HL,BC          ;Pt past patch code
2A74 F1      05860          POP     AF          ;Recover prog byte

```

```

2A75 77      05870 LD      (HL),A      ; & stuff after fix code
2A76 E1      05880 POP      HL          ;Rcvr buf ptr
2A77 23      05890 INC      HL          ;Bump & loop
2A78 18E3    05900 JR       RIPPL1
           05910 ;
2A7A FE1C    05920 RIPPL2 CP      1CH      ;Got to end of file?
2A7C C2452D  05930 JP      NZ,FILERR ;Quit on any other error
2A7F 117F2D  05940 LD      DE,PGMDCB
2A82 ED4BC62B 05950 LD      BC,(RPRMAP9+1) ;Get length of patch
2A86 7E      05960 RIPPL3 LD      A,(HL)  ;Put rest of program
2A87 23      05970 INC      HL          ; (ie the bytes = to
2A88 CD112D  05980 CALL    $PUT      ; length of patch code)
2A8B 0B      05990 DEC      BC          ;Do until len left = 0
2A8C 78      06000 LD      A,B
2A8D B1      06010 OR       C
2A8E 20F6    06020 JR      NZ,RIPPL3
2A90 E1      06030 POP      HL
2A91 C9      06040 RET
           06050 ;
           06060 ;      Routine to read & convert fix code values
           06070 ;
2A92 AF      06080 PRSFX XOR      A      ;Entry to clear
2A93 32AB2A  06090 LD      (STRFLG+1),A ; STRING check
2A96 7E      06100 PRSFX1 LD      A,(HL)  ;P/u patch char
2A97 FE03    06110 CP      ETX      ;End of text?
2A99 CA612D  06120 JP      Z,PCHERR ;Error if so
2A9C FE22    06130 CP      ""      ;String?
2A9E 200A    06140 JR      NZ,STRFLG
2AA0 32AB2A  06150 LD      (STRFLG+1),A ;Stuff string indicator
2AA3 23      06160 INC      HL
2AA4 7E      06170 LD      A,(HL)  ;P/u char
2AA5 FE03    06180 CP      ETX      ;End again?
2AA7 CA612D  06190 JP      Z,PCHERR
2AAA 3E00    06200 STRFLG LD      A,0    ;Test string flag
2AAC B7      06210 OR       A
2AAD 7E      06220 LD      A,(HL)  ;P/u char again
2AAE 23      06230 INC      HL      ;Bump pointer
2AAF C0      06240 RET      NZ      ;Ret if "" was prev char
2AB0 CDD72A  06250 CALL    CVTBIN ;Convert hex digit to bin
2AB3 4F      06260 LD      C,A      ;Save value
2AB4 7E      06270 LD      A,(HL)  ;P/u next digit
2AB5 23      06280 INC      HL
2AB6 FE03    06290 CP      ETX      ;Backup pointer and ret
2AB8 CA612D  06300 JP      Z,PCHERR ; if next char is not hex
2ABB FE30    06310 CP      '0'    ; else pack it into regC
2ABD 3815    06320 JR      C,PRSFX3 ; & place in reg A
2ABF FE3A    06330 CP      '9'+1
2AC1 3804    06340 JR      C,PRSFX2
2AC3 FE41    06350 CP      'A'
2AC5 380D    06360 JR      C,PRSFX3
2AC7 CB01    06370 PRSFX2 RLC      C      ;Assume digit, move
2AC9 CB01    06380 RLC      C      ; over a nybble
2ACB CB01    06390 RLC      C
2ACD CB01    06400 RLC      C
2ACF CDD72A  06410 CALL    CVTBIN ;Get hex digit
2AD2 B1      06420 OR       C      ;Merge hi-order nybble
2AD3 C9      06430 RET
2AD4 79      06440 PRSFX3 LD      A,C      ;Non-hex char,
2AD5 2B      06450 DEC      HL      ; rcvr & exit
2AD6 C9      06460 RET
           06470 ;

```

```

06480 ;      Routine to convert hex digit to binary
06490 ;
2AD7 D630 06500 CVTBIN SUB 30H ;1st adjustment to binary
2AD9 DA5D2D 06510 JP C, NONHEX ;Quit if too low
2ADC FE0A 06520 CP 10 ;0-9 range?
2ADE D8 06530 RET C ;Back if so
2ADF CBAF 06540 RES 5,A ;In case lower case
2AE1 D607 06550 SUB 7
2AE3 FE10 06560 CP 16 ;Less than F+1?
2AE5 D8 06570 RET C ;Ok if so
2AE6 C35D2D 06580 JP NONHEX ; else abort
06590 ;
06600 ;      Routine to find ISAM member pointer in map table
06610 ;
2AE9 117F2D 06620 FISAM LD DE,PGMDCB
2AEC CD072D 06630 FISAM1 CALL $GET1 ;Get a type byte
2AEF 2808 06640 JR Z,FISAM1A ;Go on no error
2AF1 FE1C 06650 CP 1CH ;EOF?
2AF3 CA412D 06660 JP Z,LIBERR ;Invalid library format
2AF6 C3322D 06670 JP IOERR ; else I/O error
2AF9 FE08 06680 FISAM1A CP 8 ;Start of map table?
2AFB 2820 06690 JR Z,FISAM3
2AFD FE0A 06700 CP 0AH ;End of map table?
2AFF CA3D2D 06710 JP Z,NOVRLY ;Should not be end
2B02 C5 06720 PUSH BC
2B03 4F 06730 LD C,A ;Save TYPE
2B04 CD0B2D 06740 CALL $GET ;Get block length
2B07 47 06750 LD B,A ;Set counter & read
2B08 0D 06760 DEC C
2B09 2008 06770 JR NZ,FISAM1B ;Go if not load record
2B0B CD0B2D 06780 CALL $GET ; else read 1st two
2B0E 05 06790 DEC B ; bytes & then fall thru
2B0F CD0B2D 06800 CALL $GET ; in case len=01 or 02
2B12 05 06810 DEC B
2B13 78 06820 FISAM1B LD A,B
2B14 C1 06830 POP BC
2B15 47 06840 LD B,A
2B16 CD0B2D 06850 FISAM2 CALL $GET ;Through the block
2B19 10FB 06860 DJNZ FISAM2
2B1B 18CF 06870 JR FISAM1 ;Go back for more
06880 ;
06890 ;      Found a map field
06900 ;
2B1D CD0B2D 06910 FISAM3 CALL $GET ;Get field length
2B20 47 06920 LD B,A ;Set counter
2B21 CD0B2D 06930 CALL $GET ;Get overlay #
2B24 05 06940 DEC B ;Reduce count
2B25 B9 06950 CP C ;Is this the one?
2B26 20EE 06960 JR NZ,FISAM2 ;Loop to next field
2B28 CD0B2D 06970 CALL $GET ;Get lo-order traadr
2B2B CD0B2D 06980 CALL $GET ;Get hi-order transfer
2B2E CD0B2D 06990 CALL $GET ;Get lo-order NRN
2B31 4F 07000 LD C,A ;Save in C
2B32 CD0B2D 07010 CALL $GET ;Get hi-order NRN
2B35 47 07020 LD B,A ;Save in B
2B36 CD0B2D 07030 CALL $GET ;Get byte offset
2B39 C9 07040 RET
07050 ;
07060 ;      Routine to repair the library map
07070 ;
2B3A 117F2D 07080 RPRMAP LD DE,PGMDCB ;Rewind the file

```

```

2B3D 010000 07090 LD BC,0
2B40 CDF52C 07100 CALL $POSN
2B43 210048 07110 LD HL,PGMDATA ;Pt to buffer area
2B46 CD0B2D 07120 RPRMAP1 CALL $GET ;Read the map into buf
2B49 FE0A 07130 CP 0AH ;End of table?
2B4B 2821 07140 JR Z,RPRMAP3
2B4D 77 07150 LD (HL),A ;Save type code
2B4E CD0B2D 07160 CALL $GET ;Get length
2B51 47 07170 LD B,A ;Set counter
2B52 7E 07180 LD A,(HL) ;Reget the TYPE
2B53 23 07190 INC HL ;Bump where to stuff len
2B54 3D 07200 DEC A ;Is this a load record?
2B55 70 07210 LD (HL),B ;Put length in too
2B56 23 07220 INC HL
2B57 200C 07230 JR NZ,RPRMAP2 ;Go if other type
2B59 CD0B2D 07240 CALL $GET ; else get two extra
2B5C 05 07250 DEC B ; & adjust length in
2B5D 77 07260 LD (HL),A ; case len = 01 or 02
2B5E 23 07270 INC HL
2B5F CD0B2D 07280 CALL $GET
2B62 05 07290 DEC B
2B63 23 07300 INC HL
2B64 77 07310 LD (HL),A
2B65 CD0B2D 07320 RPRMAP2 CALL $GET ;Save member # & rest of
2B68 77 07330 LD (HL),A ; data entries
2B69 23 07340 INC HL
2B6A 10F9 07350 DJNZ RPRMAP2
2B6C 18D8 07360 JR RPRMAP1
07370 ;
07380 ; Found end of table
07390 ;
2B6E 77 07400 RPRMAP3 LD (HL),A ;Show map end
2B6F 210048 07410 LD HL,PGMDATA ;Pt to beginning
2B72 7E 07420 RPRMAP4 LD A,(HL) ;P/u type code
2B73 23 07430 INC HL
2B74 46 07440 LD B,(HL) ;P/u length
2B75 23 07450 INC HL
2B76 FE08 07460 CP 8 ;Map is type 8
2B78 2811 07470 JR Z,RPRMAP6
2B7A FE0A 07480 CP 0AH ;End of map?
2B7C CA3D2D 07490 JP Z,NOVRLY ;Should not have gotten
2B7F 3D 07500 DEC A
2B80 2004 07510 JR NZ,RPRMAP5
2B82 23 07520 INC HL ;You should know what
2B83 05 07530 DEC B ; this is for by now
2B84 23 07540 INC HL
2B85 05 07550 DEC B
2B86 23 07560 RPRMAP5 INC HL ;Bypass this field
2B87 10FD 07570 DJNZ RPRMAP5
2B89 18E7 07580 JR RPRMAP4
07590 ;
07600 ; Found a type 8, check if ISAM # matches
07610 ;
2B8B 7E 07620 RPRMAP6 LD A,(HL) ;P/u member #
2B8C 23 07630 INC HL
2B8D 05 07640 DEC B ;Count down
2B8E FE00 07650 OVRLY CP 0 ;Compare to patched one
2B90 20F4 07660 JR NZ,RPRMAP5 ;Keep reading until found
2B92 23 07670 INC HL ;Bypass transfer address
2B93 23 07680 INC HL
2B94 5E 07690 LD E,(HL) ;P/u the position lo

```

```

2B95 23      07700      INC      HL
2B96 56      07710      LD        D,(HL)      ; & the pos hi
2B97 23      07720      INC      HL
2B98 4E      07730      LD        C,(HL)      ; & the byte offset
2B99 78      07740      LD        A,B          ;Calc ptr to next field
2B9A D604    07750      SUB        4
2B9C 47      07760      LD        B,A
2B9D 23      07770      INC      HL
2B9E 10FD    07780      DJNZ     $-1          ;Loop to next field
2BA0 7E      07790 RPRMAP7 LD        A,(HL)      ;End of table?
2BA1 FE0A    07800      CP        0AH          ;If end, write the
2BA3 2836    07810      JR        Z,RWRMAP      ; map back to disk
2BA5 23      07820      INC      HL          ;Pt to field length
2BA6 46      07830      LD        B,(HL)
2BA7 23      07840      INC      HL          ;Pt to member #
2BA8 23      07850      INC      HL          ;Transfer Low
2BA9 23      07860      INC      HL          ;Transfer High
2BAA 23      07870      INC      HL          ;NRN Low
2BAB 78      07880      LD        A,B          ;Adjust count for
2BAC D604    07890      SUB        4          ; 4 INC HLs
2BAE 47      07900      LD        B,A
2BAF 7E      07910      LD        A,(HL)
2BB0 23      07920      INC      HL          ;If position is the same
2BB1 BB      07930      CP        E          ; as that of patched
2BB2 200F    07940      JR        NZ,RPRMAP8    ; one, its posn has not
2BB4 7E      07950      LD        A,(HL)      ; changed, so don't
2BB5 23      07960      INC      HL          ; change it
2BB6 05      07970      DEC      B
2BB7 BA      07980      CP        D          ;Cp the hi order
2BB8 200B    07990      JR        NZ,RPRMAP9
2BBA 7E      08000      LD        A,(HL)
2BBB B9      08010      CP        C          ; and the offset
2BBC 2007    08020      JR        NZ,RPRMAP9
2BBE 23      08030 LPFLD  INC      HL
2BBF 10FD    08040      DJNZ     $-1          ;Loop to end of field
2BC1 18DD    08050      JR        RPRMAP7
                08060 ;
                08070 ;      Add the patch length to each position vector
                08080 ;
2BC3 23      08090 RPRMAP8 INC      HL          ;Bump to offset byte
2BC4 05      08100      DEC      B
2BC5 110000  08110 RPRMAP9 LD        DE,0        ;P/u patch length
2BC8 7E      08120      LD        A,(HL)      ;P/u offset & add
2BC9 83      08130      ADD      A,E          ;Lo-order patch length
2BCA 77      08140      LD        (HL),A
2BCB 2B      08150      DEC      HL          ;Pt to NRN
2BCC 2B      08160      DEC      HL
2BCD 7E      08170      LD        A,(HL)      ;P/u NRN lo-order
2BCE 8A      08180      ADC      A,D          ;Add to it
2BCF 77      08190      LD        (HL),A
2BD0 23      08200      INC      HL          ;Pt to pos hi order
2BD1 7E      08210      LD        A,(HL)      ;P/u the hi
2BD2 CE00    08220      ADC      A,0        ;Add in any carry
2BD4 77      08230      LD        (HL),A
2BD5 23      08240      INC      HL          ;Pt to next map field
2BD6 110000  08250      LD        DE,0
2BD9 18E3    08260      JR        LPFLD        ;Loop
                08270 ;
                08280 ;      Routine to re-write the library map table
                08290 ;
2BDB 117F2D  08300 RWRMAP LD        DE,PGMDCB    ;Rewind the program file

```



```

2BDE 010000 08310 LD BC,0
2BE1 CDF52C 08320 CALL $POSN
2BE4 210048 08330 LD HL,PGMDATA ;Pt to mapbuf start
2BE7 7E 08340 RWRMAP1 LD A,(HL) ;Ret when we get to
2BE8 FE0A 08350 CP 0AH ; the map end type
2BEA C8 08360 RET Z
2BEB 4F 08370 LD C,A ;Save the type
2BEC 23 08380 INC HL
2BED CD112D 08390 CALL $PUT ;Put the type
2BF0 7E 08400 LD A,(HL) ;P/u length
2BF1 23 08410 INC HL
2BF2 47 08420 LD B,A ;Save as counter
2BF3 CD112D 08430 CALL $PUT ;Put out the length
2BF6 0D 08440 DEC C ;Again, by now...
2BF7 200C 08450 JR NZ,RWRMAP2
2BF9 7E 08460 LD A,(HL)
2BFA 23 08470 INC HL
2BFB CD112D 08480 CALL $PUT
2BFE 05 08490 DEC B
2BFF 7E 08500 LD A,(HL)
2C00 23 08510 INC HL
2C01 CD112D 08520 CALL $PUT
2C04 05 08530 DEC B
2C05 7E 08540 RWRMAP2 LD A,(HL) ;Put block of code
2C06 23 08550 INC HL
2C07 CD112D 08560 CALL $PUT
2C0A 10F9 08570 DJNZ RWRMAP2
2C0C 18D9 08580 JR RWRMAP1 ;Loop for more
08590 ;
08600 ; This routine enters at PASS1. It does the first pass
08610 ; thru the fix data, and checks for parms as well
08620 ; as checking the Drr,bb and Frr,bb matches.
08630 ;
2C0E 320627 08640 SPASS2 LD (PASS2),A ;Flag pass 2
2C11 C3F726 08650 JP DOFIX ;Start over
08660 ;
2C14 FE2E 08670 PASS1 CP '.' ;Comment line?
2C16 2825 08680 JR Z,OK
2C18 FE03 08690 CP ETX ;End of fix data?
2C1A 28F2 08700 JR Z,SPASS2 ;End of pass1
2C1C CBAF 08710 RES 5,A ;Make Upper case
2C1E FE44 08720 CP 'D' ;D line patch?
2C20 281E 08730 JR Z,FCHK
2C22 FE52 08740 CP 'R' ;Remove parm?
2C24 CA3A28 08750 JP Z,REMOVE
2C27 FE4F 08760 CP 'O' ;Special 0 parm?
2C29 CAD128 08770 JP Z,OVERB
2C2C FE46 08780 CP 'F' ;Find line data?
2C2E 280D 08790 JR Z,OK
2C30 FE59 08800 CP 'Y' ;Yank parm?
2C32 2809 08810 JR Z,OK
2C34 FE4C 08820 CP 'L' ;Library ISAM number?
2C36 2805 08830 JR Z,OK
2C38 FE58 08840 CP 'X' ;X line patch?
2C3A C2612D 08850 JP NZ,PCHERR ;If not one of these, abort
2C3D C3C527 08860 OK JP COMMENT
08870 ;
08880 ; Check the Drr,bb (if Remove) or Frr,bb line
08890 ;
2C40 3EFF 08900 FCHK LD A,0FFH ;If 0 parm = OFF, then
2C41 08910 OPARM EQU $-1 ; don't do the check

```

The Source	UTILITY Files	PATCH - LS-DOS 6.2	Page 00018
2C42 B7	08920	OR A	
2C43 CAC527	08930	JP Z,COMMENT	;Skip check if 0=OFF
2C46 3E00	08940	LD A,\$-\$;Remove parm used?
2C47	08950 RPARM	EQU \$-1	
2C48 B7	08960	OR A	
2C49 C2862C	08970	JP NZ,YANKD	;Reverse D & F lines if so
2C4C 22772D	08980	LD (DL),HL	;Save D pointer
2C4F CDA02C	08990	CALL SKPLN	;Move to F line
2C52 CD582C	09000	CALL DOCHK	;Cp F line bytes w/file
2C55 C3FA26	09010	JP DOFIX1	
	09020 ;		
	09030 ;		
	09040 ;		
	09050 ;	Checks Drr,bb and Frr,bb addresses for a match	
	09060 ;	Checks Frr,bb against program file if patching, or	
	09070 ;	Drr,bb if removing	
2C58 226B2D	09080 DOCHK	LD (SETMSG+1),HL	;Set line error msg
2C5B E5	09090	PUSH HL	;Save posn
2C5C ED5B772D	09100	LD DE,(DL)	;Get D or F line
2C60 0603	09110	LD B,3	;Init check count
2C62 23	09120 CP3	INC HL	
2C63 13	09130	INC DE	
2C64 1A	09140	LD A,(DE)	
2C65 BE	09150	CP (HL)	
2C66 C2C62C	09160	JP NZ,FERROR	; 'FIND' error
2C69 10F7	09170	DJNZ CP3	;Check first 3 bytes
	09180 ;		
2C6B 0603	09190	LD B,3	;Assume was 2 digit rec #
2C6D 3E2C	09200	LD A,', '	;Comma?
2C6F BE	09210	CP (HL)	
2C70 2802	09220	JR Z,CP5	;Yes, continue
2C72 0605	09230	LD B,5	;Adjust, assume 4 dig rec #
2C74 23	09240 CP5	INC HL	;Check rest of 'rr,bb' string
2C75 13	09250	INC DE	
2C76 1A	09260	LD A,(DE)	
2C77 BE	09270	CP (HL)	
2C78 C2C62C	09280	JP NZ,FERROR	; 'FIND' error
2C7B 10F7	09290	DJNZ CP5	
	09300 ;		
2C7D E3	09310	EX (SP),HL	;Pointer to '=' in fix line
2C7E CDE327	09320	CALL DPOSN	;Posn file
2C81 E1	09330	POP HL	
2C82 CD0A28	09340	CALL DLINE	;Check line for match
2C85 C9	09350	RET	
	09360 ;		
	09370 ;	Remove used. Check Drr,bb lines instead of Frr,bb lines	
	09380 ;		
2C86 E5	09390 YANKD	PUSH HL	;Save D line pointer
2C87 CDA02C	09400	CALL SKPLN	;Move to F line
2C8A 22772D	09410	LD (DL),HL	;Save pointer
2C8D E1	09420	POP HL	;=>D line
2C8E E5	09430	PUSH HL	;Save D line again
2C8F CD582C	09440	CALL DOCHK	;Test D line
2C92 E1	09450	POP HL	;=>'D'
2C93 362E	09460	LD (HL), '.'	;Make comment for pass2
2C95 2A772D	09470	LD HL,(DL)	
2C98 3644	09480	LD (HL), 'D'	;Make 'F' line into D line
2C9A CDA02C	09490	CALL SKPLN	;=>next line
2C9D C3FA26	09500	JP DOFIX1	
	09510 ;		
	09520 ;	Skip past the current line, posn to start of next	

```

09530 ;
2CA0 CDA92C 09540 SKPLN CALL SKPLN1 ;Move past current line
2CA3 7E 09550 LD A,(HL) ;Check 1st char next line
2CA4 FE2E 09560 CP '.' ;Is it comment?
2CA6 28F8 09570 JR Z,SKPLN ;Then skip it too
2CA8 C9 09580 RET
09590 ;
2CA9 7E 09600 SKPLN1 LD A,(HL) ;P/u line char
2CAA 23 09610 INC HL
2CAB FE0D 09620 CP CR ;Physical EOL?
2CAD C8 09630 RET Z
2CAE FE3B 09640 CP ';' ;Logical EOL?
2CB0 C8 09650 RET Z
2CB1 18F6 09660 JR SKPLN1 ;Loop until EOL
09670 ;
09680 ; Get the next char, convert to UC
09690 ;
2CB3 7E 09700 GETNXT LD A,(HL) ;P/u the char
2CB4 23 09710 INC HL ;Bump the buffer ptr
2CB5 CBAF 09720 RES 5,A ;Convert to upper
2CB7 C9 09730 RET
09740 ;
09750 ; Either write a char or check for a match
09760 ;
2CB8 4F 09770 PUTORCHK LD C,A ;Char in question
2CB9 3A0627 09780 LD A,(PASS2) ;Write pass?
2CBC B7 09790 OR A
2CBD 79 09800 LD A,C ;Char back in A
2CBE C2112D 09810 JP NZ,$PUT ;Writing patch..
2CC1 CD0B2D 09820 CALL $GET ;Get next char fm file
2CC4 B9 09830 CP C ;Match w/patch?
2CC5 C8 09840 RET Z ;OK if match
2CC6 211130 09850 FERROR LD HL,LOCERR$ ;Init "Find mismatch
2CC9 C3642D 09860 JP ERRDSP ;Dsply and quit
09870 ;
09880 ; Count patch lines
09890 ;
2CCC E5 09900 CNTLIN PUSH HL
2CCD 2A742D 09910 LD HL,(LINCNT) ;Get current count,
2CD0 23 09920 INC HL ; += 1
2CD1 22742D 09930 LD (LINCNT),HL ; and put it back
2CD4 E1 09940 POP HL
2CD5 C9 09950 RET
09960 ;
09970 ; After an error, show file not closed if needed
09980 ;
2CD6 3A762D 09990 FLOPN LD A,(WRFLAG) ;Did we modify file?
2CD9 B7 10000 OR A
2CDA 2007 10010 JR NZ,MESS ;Yes, don't close it
2CDC 10020 @@CLOSE ;No changes
2CDC 3E3C 00035 LD A,60
2CDE EF 00036 RST 40
2CDF C2322D 10030 JP NZ,IOERR
2CE2 C9 10040 RET
2CE3 212430 10050 MESS LD HL,WARN1$ ;File is modified but
2CE6 CD1F2D 10060 CALL $DSPLY ;PATCH did not complete
2CE9 214B30 10070 LD HL,WARN2$ ; "oops..."
2CEC C31F2D 10080 JP $DSPLY ;Then return to caller
10090 ;
2CEF 10100 $OPEN @@OPEN
2CEF 3E3B 00037 LD A,59

```

```

2CF1 EF      00038      RST      40
2CF2 203E    10110      JR        NZ,IOERR
2CF4 C9      10120      RET
2CF5         10130 $POSN  @@POSN
2CF5 3E42    00039      LD        A,66
2CF7 EF      00040      RST      40
2CF8 2038    10140      JR        NZ,IOERR
2CFA C9      10150      RET
2CFB         10160 $BKSP  @@BKSP
2CFB 3E3D    00041      LD        A,61
2CFD EF      00042      RST      40
2CFE 2032    10170      JR        NZ,IOERR
2D00 C9      10180      RET
2D01         10190 $RWRT  @@RWRT
2D01 3E46    00043      LD        A,70
2D03 EF      00044      RST      40
2D04 202C    10200      JR        NZ,IOERR
2D06 C9      10210      RET
2D07         10220 $GET1  @@GET
2D07 3E03    00045      LD        A,3
2D09 EF      00046      RST      40
2D0A C9      10230      RET
2D0B         10240 $GET   @@GET
2D0B 3E03    00047      LD        A,3
2D0D EF      00048      RST      40
2D0E 2022    10250      JR        NZ,IOERR
2D10 C9      10260      RET
2D11 C5      10270 $PUT   PUSH     BC
2D12 4F      10280      LD        C,A
2D13 3EFF    10290      LD        A,0FFH
2D15 32762D  10300      LD        (WRFLAG),A
2D18         10310      @@PUT
2D18 3E04    00049      LD        A,4
2D1A EF      00050      RST      40
2D1B C1      10320      POP      BC
2D1C 2014    10330      JR        NZ,IOERR
2D1E C9      10340      RET
2D1F         10350 $DSPLY @@DSPLY
          00051      IFEQ     00H,1
          00052      LD        HL,
          00053      ENDIF
2D1F 3E0A    00054      LD        A,10
2D21 EF      00055      RST      40
2D22 200E    10360      JR        NZ,IOERR
2D24 C9      10370      RET
2D25         10380 $READ  @@READ
2D25 3E43    00056      LD        A,67
2D27 EF      00057      RST      40
2D28 2008    10390      JR        NZ,IOERR
2D2A C9      10400      RET
2D2B C5      10410 $DSP   PUSH     BC
2D2C 4F      10420      LD        C,A
2D2D         10430      @@DSP
2D2D 3E02    00058      LD        A,2
2D2F EF      00059      RST      40
2D30 C1      10440      POP      BC
2D31 C8      10450      RET      Z
          10460 ;
          10470 ;
          10480 ;
          10490 IOERR  LD        L,A
          ;HL also gets error #
;Use this one if prog might get EOF
;This one if EOF is also error
;Flag..
;That file is modified
;If OK else fall error
Error handling

```

```

2D33 2600      10500      LD      H,0
2D35 F6C0      10510      OR       0C0H      ;Abbrev, return
2D37 4F        10520      LD       C,A
2D38          10530      @@ERROR      ;Display the error
2D38 3E1A      00060      LD       A,26
2D3A EF        00061      RST      40
2D3B 181D      10540      JR       QUIT1
                10550      ;
                10560      ;      Internal error routine
                10570      ;
2D3D 211B2F    10580      NOVRLY  LD      HL,NOVRLY$      ;"Library not found
2D40 DD        10590      DB      0DDH
2D41 21352F    10600      LIBERR  LD      HL,LIBERR$      ;"Invalid library
2D44 DD        10610      DB      0DDH
2D45 21652F    10620      FILERR  LD      HL,FILERR$      ;"Not load file format
2D48 DD        10630      DB      0DDH
2D49 215F2E    10640      PRMERR  LD      HL,PRMERR$      ;"Parm error
2D4C DD        10650      DB      0DDH
2D4D 21CE2F    10660      TOOBIG  LD      HL,TOOBIG$      ;"Fix file too big
2D50 DD        10670      DB      0DDH
2D51 21442E    10680      PGMREQ  LD      HL,PGMREQ$      ;"Patch what file?
2D54          10690      ERREXIT @@LOGOT      ;Display the error
                00062      IFEQ    00H,1
                00063      LD      HL,
                00064      ENDIF
2D54 3E0C      00065      LD      A,12
2D56 EF        00066      RST      40
2D57 21FFFF    10700      LD      HL,-1      ;Set abort code
2D5A C3AE27    10710      QUIT1   JP      $QUIT
                10720      ;
2D5D 217C2F    10730      NONHEX  LD      HL,NONHEX$      ;"Non hex digit
2D60 DD        10740      DB      0DDH
2D61 214C2F    10750      PCHERR  LD      HL,PCHERR$      ;"Patch format error
2D64 E5        10760      ERRDSP  PUSH    HL
2D65 3E0D      10770      LD      A,CR      ;Move the cursor down
2D67 CD2B2D    10780      CALL    $DSP
2D6A 210000    10790      SETMSG  LD      HL,0
2D6D          10800      @@LOGOT
                00067      IFEQ    00H,1
                00068      LD      HL,
                00069      ENDIF
2D6D 3E0C      00070      LD      A,12
2D6F EF        00071      RST      40
2D70 E1        10810      POP     HL
2D71 18E1      10820      JR      ERREXIT
                10830      ;
2D73 00        10840      YNKFLG  DB      0      ;Was function YANK?
2D74 0000      10850      LINCNT  DW      0      ;Count lines installed
2D76 00        10860      WRFLAG  DB      0      ;Did pgm write to file?
2D77 0000      10870      DL      DW      0      ;Save pointer to line
2D79 43        10880      CMDEXT  DB      'CMD'
                4D 44
2D7C 46        10890      FIXEXT  DB      'FIX'
                49 58
2D7F 00        10900      PGMDCB  DB      0
0020          10910      DS      32
0020          10920      FIXDCB  DS      32
2DC0 50        10930      HELLO$  DB      'PATCH'
                41 54 43 48
2DC5          10940      *GET     CLIENT:3
                10950      ;CLIENTS/ASM - File to establish sign-on headers

```

```

                10960 ;
2DC5 20        10970 DB
        2D 20 36 2E 32 2E 30 20
        2D 20 43 6F 70 79 72 69
        67 68 74 20 31 39 38 32
        2F 38 33 2F 38 34 20 62
        79 20 4C 6F 67 69 63 61
        6C
2DEF 20        10980 DB
        53 79 73 74 65 6D 73 2C
        20 49 6E 63 2E 20 20 20
        20 20 20 0A
                10990 ;
2E04 41        11000 DB
        6C 6C 20 52 69 67 68 74
        73 20 52 65 73 65 72 76
        65 64 2E 20 4C 69 63 65
        6E 73 65 64 20 31 39 38
        32 2F 38 33 2F 38 34
2E2C 20        11010 DB
        74 6F 20 78 78 78 78 78
        78 78 78 78 78 78 78 78
        78 78 78 78 78 0A 0D
2E44 50        11020 PGMREQ$ DB
        52 4F 47 52 41 4D 20 66
        69 6C 65 20 6E 61 6D 65
        20 72 65 71 75 69 72 65
        64 0D
2E5F 50        11030 PRMERR$ DB
        61 72 61 6D 65 74 65 72
        20 65 72 72 6F 72 0D
2E6F 1D        11040 POSLD$ DB
        50 6F 73 69 74 69 6F 6E
        69 6E 67 20 6C 6F 61 64
        20 66 69 6C 65 1E 20 03
2E88 1D        11050 RDGINP$ DB
        52 65 61 64 69 6E 67 20
        69 6E 70 75 74 1E 20 03
2E99 1D        11060 GENPCH$ DB
        47 65 6E 65 72 61 74 69
        6E 67 20 70 61 74 63 68
        1E 20 03
2EAD 1D        11070 INSPCH$ DB
        49 6E 73 74 61 6C 6C 69
        6E 67 20 70 61 74 63 68
        1E 20 03
2EC1 1D        11080 BLDMP$ DB
        52 65 2D 62 75 69 6C 64
        69 6E 67 20 6C 69 62 72
        61 72 79 20 6D 61 70 1E
        20 03
2EDC 1D        11090 YNKPCH$ DB
        59 61 6E 6B 69 6E 67 20
        70 61 74 63 68 20 66 72
        6F 6D 20 66 69 6C 65 1E
        20 03
2EF7 0A        11100 NOYANK$ DB
        43 61 6E 27 74 20 79 61
        6E 6B 2C 20
2F04 70        11110 DB
        61 74 63 68 20 6E 6F 74

```

```

' - 6.2.0 - Copyright 1982/83/84 by Logical'

' Systems, Inc. ',10

'All Rights Reserved. Licensed 1982/83/84'

' to xxxxxxxxxxxxxxxxxxxxxx',10,13

'PROGRAM file name required',CR

'Parameter error',CR

29,'Positioning load file',30,32,3

29,'Reading input',30,32,3

29,'Generating patch',30,32,3

29,'Installing patch',30,32,3

29,'Re-building library map',30,32,3

29,'Yanking patch from file',30,32,3

LF,'Can't yank, '

'patch not in load file',CR

```

20 69 6E 20 6C 6F 61 64	
20 66 69 6C 65 0D	
2F1B 4C 11120 NOVRLY\$ DB	'Library overlay not found',CR
69 62 72 61 72 79 20 6F	
76 65 72 6C 61 79 20 6E	
6F 74 20 66 6F 75 6E 64	
0D	
2F35 49 11130 LIBERR\$ DB	'Invalid library format',CR
6E 76 61 6C 69 64 20 6C	
69 62 72 61 72 79 20 66	
6F 72 6D 61 74 0D	
2F4C 50 11140 PCHERR\$ DB	'Patch input format error',CR
61 74 63 68 20 69 6E 70	
75 74 20 66 6F 72 6D 61	
74 20 65 72 72 6F 72 0D	
2F65 4C 11150 FILERR\$ DB	'Load file format error',CR
6F 61 64 20 66 69 6C 65	
20 66 6F 72 6D 61 74 20	
65 72 72 6F 72 0D	
2F7C 4E 11160 NONHEX\$ DB	'Non-hex digit encountered',CR
6F 6E 2D 68 65 78 20 64	
69 67 69 74 20 65 6E 63	
6F 75 6E 74 65 72 65 64	
0D	
2F96 0A 11170 SUCCES\$ DB	LF,'Patch function completed.',CR
50 61 74 63 68 20 66 75	
6E 63 74 69 6F 6E 20 63	
6F 6D 70 6C 65 74 65 64	
2E 0D	
2FB1 20 11180 LINMSG\$ DB	' No patch line'
20 20 4E 6F 20 70 61 74	
63 68 20 6C 69 6E 65	
2FC1 73 11190 PLURAL DB	's installed.',CR
20 69 6E 73 74 61 6C 6C	
65 64 2E 0D	
2FCE 46 11200 TOOBIG\$ DB	'Fix file too big - partition it',CR
69 78 20 66 69 6C 65 20	
74 6F 6F 20 62 69 67 20	
2D 20 70 61 72 74 69 74	
69 6F 6E 20 69 74 0D	
2FEE 50 11210 YANKMSG DB	'Patch successfully yanked',CR
61 74 63 68 20 73 75 63	
63 65 73 73 66 75 6C 6C	
79 20 79 61 6E 6B 65 64	
0D	
3008 03 11220 NAMLEN\$ DB	3 ;Length of fix file name
3009 43 11230 NAMFIX\$ DB	'CLP ' ;Fix file name
4C 50 20 20 20 20	
3011 46 11240 LOCERR\$ DB	'FIND line mismatch',CR
49 4E 44 20 6C 69 6E 65	
20 6D 69 73 6D 61 74 63	
68 0D	
3024 57 11250 WARN1\$ DB	'WARNING - File '
41 52 4E 49 4E 47 20 2D	
20 46 69 6C 65 20	
3033 20 11260 FNMS\$ DB	'
20 20 20 20 20 20 20 20	
20 20 20 20 20 20 20 20	
20 20 20 20 20 20 20 20	
304B 20 11270 WARN2\$ DB	' Not Closed',CR
4E 6F 74 20 43 6C 6F 73	

```

        65 64 0D
3057 80      11280 ;
3058 56      11290 PTBL$ DB 80H
3059 52      11300 DB FLAG!ABB!6
        45 4D 4F 56 45 00 'REMOVE',0
3060 8D26    11320 DW RPARAM1
3062 41      11330 DB FLAG!1
3063 4F      11340 DB '0',0
        00
3065 9426    11350 DW OPARAM1
3067 00      11360 NOP
        11370 ;
3100      11380 ORG $<-8+1<+8
0100      11390 FIXBUF DS 256 ;I/O buffer for /FIX
0100      11400 LIBBUF DS 256 ;I/O buffer for ISAM
0100      11410 PGMBUF DS 256 ;I/O buffer for PGM
1400      11420 FIXDATA DS 1400H ;5k allotted for fix data
4800      11430 PGMDATA EQU $ ;Takes the rest of core
        05100 ;
2600      05110 END BEGIN

```


\$BKSP	2CFB \$DSP	2D2B \$DSPLY	2D1F
\$GET	2D0B \$GET1	2D07 \$OPEN	2CEF
\$POSN	2CF5 \$PUT	2D11 \$QUIT	27AE
\$READ	2D25 \$RWRIT	2D01 @@1	0000
@@2	0000 @@3	0000 @@4	0000
@MOD2	0000 @MOD4	FFFF ABB	0010
BEGIN	2600 BEGINA	2609 BLDMAP\$	2EC1
CKLIN	2674 CKLIN1	26A0 CKLIN2	26B1
CKLIN3	26B5 CMDEXT	2D79 CNTLIN	2CCC
COMMENT	27C5 CP3	2C62 CP5	2C74
CR	000D CVTBIN	2AD7 DL	2D77
DLINE	280A DOCHK	2C58 DOFIX	26F7
DOFIX1	26FA DOXVB	29C4 DOXVB1	29EB
DOXVB2	29EE DOXVB3	29F3 DOXVB4	2A0A
DOXVB5	2A12 DOXVB6	2A29 DPOSN	27E3
DVERB	27D7 DVERB1	27F3 DVERB2	2810
DVERB3	2811 DVERB4	282C DVERB4A	282D
EOL1	27D4 ERRDSP	2D64 ERREXIT	2D54
ETX	0003 EXLOG	27A8 FCHK	2C40
FERROR	2CC6 FILERR	2D45 FILERR\$	2F65
FISAM	2AE9 FISAM1	2AEC FISAM1A	2AF9
FISAM1B	2B13 FISAM2	2B16 FISAM3	2B1D
FIXBUF	3100 FIXDATA	3400 FIXDCB	2DA0
FIXEXT	2D7C FLAG	0040 FLOPN	2CD6
FXM\$	3033 FXNAM	2658 FXNAM1	266A
FXNAM2	266F GENPCH\$	2E99 GETNXT	2CB3
HELLO\$	2DC0 INSPCH\$	2EAD IOERR	2D32
LF	000A LIBBUF	3200 LIBERR	2D41
LIBERR\$	2F35 LINCNT	2D74 LINMSG\$	2FB1
LOCERR\$	3011 LPFLD	2BBE LVERB	2900
MESS	2CE3 NAMFIX\$	3009 NAMLEN\$	3008
NOCHG	27A5 NONHEX	2D5D NONHEX\$	2F7C
NOVRLY	2D3D NOVRLY\$	2F1B NOYANK\$	2EF7
NTONE	279F OISOFF	28F8 OISON	28F7
OK	2C3D OPARM	2C41 OPARM1	2694
OVERB	28D1 OVRLY	2B8E PASS1	2C14
PASS2	2706 PCHDUN	276D PCHERR	2D61
PCHERR\$	2F4C PGMBUF	3300 PGMDATA	4800
PGMDCB	2D7F PGMREQ	2D51 PGMREQ\$	2E44
PLURAL	2FC1 POSFIL	294D POSFIL1	2957
POSFIL2	297B POSLD\$	2E6F PRMERR	2D49
PRMERR\$	2E5F PRSFX1	2A92 PRSFX1	2A96
PRSFX2	2AC7 PRSFX3	2AD4 PTBL\$	3057
PUTORCHK	2CB8 QUIT1	2D5A RDFIX	26BE
RDFIX1	26DB RDFIX2	26EF RDFIX3	26F4
RDGINP\$	2E88 REMOVE	283A RIPPL1	2A5D
RIPPL2	2A7A RIPPL3	2A86 RIPPLE	2A2F
RPARM	2C47 RPARM1	268D RPRMAP	2B3A
RPRMAP1	2B46 RPRMAP2	2B65 RPRMAP3	2B6E
RPRMAP4	2B72 RPRMAP5	2B86 RPRMAP6	2B8B
RPRMAP7	2BA0 RPRMAP8	2BC3 RPRMAP9	2BC5
RWRMAP	2BDB RWRMAP1	2BE7 RWRMAP2	2C05
SETMSG	2D6A SKPLN	2CA0 SKPLN1	2CA9
SPASS2	2C0E STACK	27BF STRFLG	2AAA
STUFNM	2982 STUFNM1	2999 STUFNM2	299F
STUFNM3	29A0 STUFNM4	29B7 SUCCES\$	2F96
TOOBIG	2D4D TOOBIG\$	2FCE TYPCOD	2869
WARN1\$	3024 WARN2\$	304B WHATIS	28F4
WRFLAG	2D76 YANK	2842 YANK1	2858
YANK2	2876 YANK3	2879 YANK4	287D

YANK5	288A YANK6	2896 YANK7	28B6
YANK8	28BD YANK9	28C6 YANKD	2C86
YANKMSG	2FEE YNKFLG	2D73 YNKPCH\$	2EDC
@@ABORT	A088 @@ADTSK	A11B @@BANK	A633
@@BKSP	A313 @@BREAK	A649 @@CHNIO	A073
@@CKBRKC	A697 @@CKDRV	A16F @@CKEOF	A328
@@CKTSK	A106 @@CLOSE	A2FE @@CLS	A681
@@CMNDI	A0B2 @@CMNDR	A0C7 @@CTL	9ED7
@@DATE	A049 @@DCSTAT	A1AE @@DEBUG	A0F1
@@DECHEX	A5B3 @@DIRRD	A520 @@DIRWR	A535
@@DIV16	A59E @@DIV8	A589 @@DODIR	A184
@@DSP	9E9B @@DSPLY	9F3B @@ERROR	A0DC
@@EXIT	A09D @@FEXT	A48D @@FLAGS	A61D
@@FNAME	A4A2 @@FSPEC	A478 @@GATRD	A50B
@@GATWR	A54A @@GET	9EAF @@GTDCB	A4CC
@@GTDCT	A4B7 @@GTMOD	A4E1 @@HDFMT	A256
@@HEX16	A5F2 @@HEX8	A5DD @@HEXDEC	A5C8
@@HIGH\$	A607 @@INIT	A2D4 @@KBD	9F13
@@KEY	9E87 @@KEYIN	9F27 @@KLTSK	A15A
@@LOAD	A44E @@LOC	A33D @@LOF	A352
@@LOGGER	9F72 @@LOGOT	9F87 @@MSG	9FBE
@@MUL16	A574 @@MUL8	A55F @@OPEN	A2E9
@@PARAM	A034 @@PAUSE	A01F @@PEOF	A367
@@POSN	A37C @@PRINT	9FD3 @@PRT	9EEB
@@PUT	9EC3 @@RAMDIR	A199 @@RDSEC	A22C
@@RDSSC	A4F6 @@READ	A391 @@REMOV	A2BF
@@RENAM	A2AA @@REW	A3A6 @@RMTSK	A130
@@RPTSK	A145 @@RREAD	A3BB @@RSLCT	A217
@@RSTOR	A1D8 @@RUN	A463 @@RWRTIT	A3D0
@@SEEK	A202 @@SEEKSC	A3E5 @@SKIP	A3FA
@@SLCT	A1C3 @@STEPI	A1ED @@TIME	A05E
@@VDCTL	A00A @@VER	A40F @@VRSEC	A241
@@WEOF	A424 @@WHERE	9EFF @@WRITE	A439
@@WRSEC	A26B @@WRSSC	A280 @@WRTRK	A295

2600 is the transfer address
 00000 Total errors

NOTES:

REPAIR/CMD - Repair a disk directory cylinder

The Repair utility will write the directory cylinder with the proper data address mark, and update certain information in the GAT that is needed by LS-DOS. Its main use is to make Model I TRSDOS disks readable by LS-DOS/TRSDOS 6.

```

00100 ;REPAIR/ASM - Directory Track Repair Program
0000 00110 TITLE <REPAIR - LS-DOS 6.2>
00120 ;
000A 00130 LF EQU 10
000D 00140 CR EQU 13
4296 00150 BLNKMPW EQU 4296H
0040 00160 FLAG EQU 01000000B
0010 00170 ABB EQU 00010000B
00180 ;
0000 00190 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVC MAC/ASM - ,LS-DOS Version VI
00020 *LIST OFF
03900 *LIST ON
0000 00200 *GET COPYCOM:3 ;Copyright message
03920 ; COPYCOM - File for Copyright COMMENT block
03930 ;
0000 03940 COM '<*(C) 1982,83,84 by LSI*>'
00210 ;
2600 00220 ORG 2600H
00230 BEGIN
2600 00240 @@CKBRKC ;Check for break
2600 3E6A 00001 LD A,106
2602 EF 00002 RST 40
2603 2804 00250 JR Z,BEGINA ;Continue if not
2605 21FFFF 00260 LD HL,-1 ; else abort
2608 C9 00270 RET
00280 ;
00290 BEGINA
2609 ED731C26 00300 LD (STACK),SP ;Save entry stack
260D E5 00310 PUSH HL ;Save ptr to CMD buffer
260E 214A28 00320 LD HL,HELLO$ ;Display the signon
2611 CD9F27 00330 CALL $DSPY
2614 E1 00340 POP HL
2615 CD2226 00350 CALL PGRM ;Normal exit is via RET
00360 ;
00370 ; Set exit condition..
00380 ;
2618 210000 00390 $EXIT LD HL,0 ;Init for no error
261B 310000 00400 QUIT$ LD SP,$-$ ;P/u original stack
261C 00410 STACK EQU $-2
261E 00420 @@CKBRKC ;Clear break before exit
261E 3E6A 00003 LD A,106
2620 EF 00004 RST 40
2621 C9 00430 RET
00440 ;
2622 7E 00450 PGRM LD A,(HL) ;Ck for drive entered
2623 FE3A 00460 CP ':' ;Colon indicator?
2625 C2E127 00470 JP NZ,PRMERR ;Quit if not
2628 23 00480 INC HL ;Point to drive #
2629 7E 00490 LD A,(HL) ;P/u drive
262A D630 00500 SUB '0' ;Cvrt to binary
262C FE08 00510 CP 8 ;Bigger than 7?
262E D24528 00520 JP NC,ILLEG ;Quit if so
00530 ;
2631 B7 00540 OR A ;Can't be drive 0
2632 CADD27 00550 JP Z,NOT0
2635 321C27 00560 LD (DRIVE),A ;Stuff for later
2638 23 00570 INC HL ;Bump past the drive
2639 4F 00580 LD C,A
263A 00590 @@GTDCT ;What's its DCT$
263A 3E51 00005 LD A,81

```

```

263C EF      000006      RST      40
              006000 ;
              006100 ;      Get any parameters
              006200 ;

263D 114E29  006300      LD      DE,PRMTBL$      ;Pt to parm table
2640          006400      @@PARAM
2640 3E11     000007      LD      A,17
2642 EF       000008      RST      40
2643 C2E127   006500      JP      NZ,PRMERR      ;Exit on parm error
2646 3A5329   006600      LD      A,(MRSP)      ;MPW parameter entered?
2649 B7        006700      OR      A
264A C20627   006800      JP      NZ,MPARM      ;Go if so
264D FDCB035E 006900      BIT     3,(IY+3)      ;Can't "repair" a hard drive
2651 C2D927   007000      JP      NZ,NIXHARD      ; except for MPW parm
2654 FDCB0466 007100      BIT     4,(IY+4)      ;If not alien controller
2658 CC1128   007200      CALL    Z,CKDRV      ; make sure disk present
265B 110000    007300      LD      DE,0      ;Read BOOT to get dir cyl
265E CDB827   007400      CALL    RDSEC
2661 AF        007500      XOR      A
2662 32002A    007600      LD      (BUF1),A      ;Set 1st byte to zero
2665 3A022A    007700      LD      A,(BUF1+2)      ;P/u the dir cyl
2668 E67F      007800      AND     7FH      ;Strip bit 7
266A 32022A    007900      LD      (BUF1+2),A      ;Put it back
266D F5        008000      PUSH    AF      ;Save dir cyl
266E CDB227   008100      CALL    WRSEC      ;Rewrite the BOOT
2671 1C        008200      INC     E
2672 CDB827   008300      CALL    RDSEC      ;Get sect 1 also
2675 F1        008400      POP     AF
2676 32022A    008500      LD      (BUF1+2),A      ;Update dir cyl
2679 F5        008600      PUSH    AF
267A CDB227   008700      CALL    WRSEC      ;Write back
267D F1        008800      POP     AF      ;Dir cyl again
              008900 ;

267E 57        009000      LD      D,A
267F 1E00      009100      LD      E,0
2681 FD7709    009200      LD      (IY+9),A      ;Set as dir cyl
2684 CDB827   009300      CALL    RDSEC      ;Read the GAT
              009400 ;

2687 FDCB04AE 009500      RES     5,(IY+4)      ;Show single sided
268B 2ECB      009600      LD      L,0CBH      ;Pt to version # byte
268D 7E        009700      LD      A,(HL)      ;Pick it up
268E FE40      009800      CP      40H      ;Earlier than a 4.0?
2690 380E      009900      JR      C,LC      ;Bypass 2 sided ck if so
2692 FE70      010000      CP      70H      ;"Later" than 6.x?
2694 300A      010100      JR      NC,LC      ;Again, no sides ck
2696 2ECD      010200      LD      L,0CDH      ;Point to CONFIG byte
2698 CB6E      010300      BIT     5,(HL)      ;Check 2-sided
269A 2804      010400      JR      Z,LC      ;Go if not
269C FDCB04EE 010500      SET     5,(IY+4)      ; else update DCT
              010600 ;

26A0 2EBF      010700      LD      L,0BFH      ;Pt to end of lockout
26A2 0660      010800      LD      B,96      ;Max cylinder count
26A4 7E        010900      LD      A,(HL)      ;P/u a lockout byte
26A5 3C        011000      INC     A      ;Locked out?
26A6 2003      011100      JR      NZ,ALIEN2      ;Exit when in use
26A8 2D        011200      DEC     L      ;Backup by 1
26A9 10F9      011300      DJNZ    ALIEN1
26AB 3EDD      011400      LD      A,-35      ;What's in use?
26AD 80        011500      ADD     A,B      ;Convert to excess
26AE 2ECC      011600      LD      L,0CCH
26B0 77        011700      LD      (HL),A      ;Stuff into GAT

```

```

01180 ;
01190 ; Construct config byte
01200 ;
26B1 FD7E04 01210 LD A,(IY+4) ;P/u # sides
26B4 E6A0 01220 AND 80H!20H
26B6 47 01230 LD B,A ;Save tempy
26B7 FD7E03 01240 LD A,(IY+3) ;P/u density
26BA E640 01250 AND 40H
26BC B0 01260 OR B ;Merge with previous
26BD 47 01270 LD B,A
26BE FD7E08 01280 LD A,(IY+8) ;P/u # grans/track
26C1 07 01290 RLCA
26C2 07 01300 RLCA ; to bits 0-2
26C3 07 01310 RLCA
26C4 E607 01320 AND 7 ;Mask off the rest
26C6 B0 01330 OR B ;Merge with previous
26C7 2C 01340 INC L ;Pt to config byte in GAT
26C8 47 01350 LD B,A ;Save for a moment
26C9 7E 01360 LD A,(HL) ;P/u present config byte
26CA E680 01370 AND 80H ;Keep only bit 7
26CC B0 01380 OR B ;Pick up the rest
26CD 77 01390 LD (HL),A ; & stuff
26CE 2E00 01400 LD L,0
26D0 CDA527 01410 CALL WRSYS ;Write the GAT
01420 ;
01430 ; Operate on the HIT
01440 ;
26D3 1C 01450 INC E ;Bump sector ptr to 1
26D4 CDB827 01460 CALL RDSEC ;Read the HIT
26D7 2C 01470 INC L ;Pt to DIR/SYS dec
26D8 36C4 01480 LD (HL),0C4H ;"correct" DEC code
26DA 2D 01490 DEC L
26DB CDA527 01500 CALL WRSYS ;Write out the HIT
26DE 0608 01510 LD B,8 ;Init for 8 sectors
26E0 1C 01520 ALIEN3 INC E ;Bump to next sector
26E1 CDB827 01530 CALL RDSEC ;Get the sector
26E4 CD8F27 01540 CALL UNOPEN ;Reset file open bit
26E7 7B 01550 LD A,E ;If DIR/SYS sector,
26E8 FE03 01560 CP 3 ; then update count & it
26EA 200E 01570 JR NZ,ALIEN4
26EC E5 01580 PUSH HL
26ED 219642 01590 LD HL,BLNKMPW ;Set DIR/SYS password
26F0 22122A 01600 LD (BUF1+12H),HL ;To blanks
26F3 3A142A 01610 LD A,(BUF1+20) ;P/u ERN of DIR/SYS
26F6 D603 01620 SUB 3 ;Account for 1st 3 done
26F8 47 01630 LD B,A ;Update loop counter
26F9 E1 01640 POP HL
26FA CDA527 01650 ALIEN4 CALL WRSYS ;Write back the sector
26FD 10E1 01660 DJNZ ALIEN3
01670 ;
26FF 01680 @@LOGOT ALCA0$ ;Advise complete - now readable
00009 IFEQ 01H,1
26FF 21CF28 00010 LD HL,ALCA0$
00011 ENDIF
2702 3E0C 00012 LD A,12
2704 EF 00013 RST 40
2705 C9 01690 RET ;Done
01700 ;
01710 ; MPW parameter to change disk password on hard drive
01720 ;
2706 110000 01730 MPARM LD DE,0 ;P/u MPW string address

```

The Source	UTILITY Files	REPAIR - LS-DOS 6.2	Page 00004
2709 CB6F	01740	BIT 5,A	;If not string, then error
270B CAE127	01750	JP Z,PRMERR	
270E FDCB035E	01760	BIT 3,(IY+3)	;Can't do if not hard
2712 CAE127	01770	JP Z,PRMERR	
2715 CD2D27	01780	CALL GETMPW	;Get and hash the entry
2718 C2C227	01790	JP NZ,IOERR	
271B 0E00	01800	LD C,0	;Init to drive requested
271C	01810	DRIVE EQU \$-1	
271D CDEE27	01820	CALL GATRD	;Read GAT into BUF1
2720 C2C227	01830	JP NZ,IOERR	;Back on error
2723 22CE2A	01840	LD (BUF1+0CEH),HL	;Stuff PW
2726 CDEF27	01850	CALL GATWR	;Write sector 0 from buf
2729 C2C227	01860	JP NZ,IOERR	;Jump on write error
272C C9	01870	RET	;Finished with Repair
	01880 ;		
	01890 ;	Enter SYS2 & hash the password	
	01900 ;		
272D CD3427	01910	GETMPW CALL GMPW1	;Get MPW into buffer
2730 C0	01920	RET NZ	
2731 3EE4	01930	LD A,0E4H	;Hash password (DE) to HL
2733 EF	01940	RST 28H	;Ret to what called
	01950 ;		
	01960 ;	Place entered password into buffer	
	01970 ;		
2734 215729	01980	GMPW1 LD HL,PSWDBUF	;Point to buffer
2737 E5	01990	PUSH HL	
2738 0608	02000	LD B,8	;Init for 8 chars
273A 1A	02010	GMPW2 LD A,(DE)	;P/u a char
273B FE0D	02020	CP CR	;End of line?
273D 280F	02030	JR Z,GMPW4	
273F FE2C	02040	CP ','	;Comma separator?
2741 280B	02050	JR Z,GMPW4	
2743 FE22	02060	CP '"'	;Closing quote?
2745 2807	02070	JR Z,GMPW4	
2747 13	02080	INC DE	;Bump input pointer
2748 77	02090	LD (HL),A	;Transfer character
2749 23	02100	INC HL	;Bump output pointer
274A 10EE	02110	DJNZ GMPW2	;Loop until done
274C 1805	02120	JR CKMPW	
274E 3620	02130	GMPW4 LD (HL),' '	;Buffer with
2750 23	02140	INC HL	; trailing spaces
2751 10FB	02150	DJNZ GMPW4	
	02160 ;		
	02170 ;	Convert to upper case and check validity	
	02180 ;		
2753 E1	02190	CKMPW POP HL	;Recover buffer start
2754 E5	02200	PUSH HL	
2755 0608	02210	LD B,8	
2757 7E	02220	LD A,(HL)	;P/u 1st char
2758 180E	02230	JR CKMPW2	; & check <A-Z>
275A 23	02240	CKMPW1 INC HL	
275B 7E	02250	LD A,(HL)	
275C FE20	02260	CP ' '	;Got to a space?
275E 2823	02270	JR Z,CKMPW7	
2760 FE30	02280	CP '0'	;Less than '0' is error
2762 3823	02290	JR C,INVMPW	
2764 FE3A	02300	CP '9'+1	<0-9> is okay for 2-n
2766 3812	02310	JR C,CKMPW3	
2768 FE41	02320	CKMPW2 CP 'A'	;Less than "A" is error
276A 381B	02330	JR C,INVMPW	
276C FE5B	02340	CP 'Z'+1	<A-Z> is okay


```

276E 380A      02350      JR      C,CKMPW3
2770 FE61      02360      CP      'a'                ;<a-z> convert to
2772 3813      02370      JR      C,INVMPW
2774 FE7B      02380      CP      'z'+1
2776 300F      02390      JR      NC,INVMPW
2778 CBAE      02400      RES      5,(HL)            ; upper case
277A 10DE      02410 CKMPW3 DJNZ    CKMPW1
277C D1        02420 CKMPW4 POP     DE                ;Point to buffer start
277D AF        02430      XOR     A
277E C9        02440      RET
277F 23        02450 CKMPW5 INC     HL
2780 BE        02460      CP      (HL)            ;No imbedded spaces
2781 2004      02470      JR      NZ,INVMPW
2783 10FA      02480 CKMPW7 DJNZ    CKMPW5
2785 18F5      02490      JR      CKMPW4
2787 211C29    02500 INVMPW LD      HL,BADMPW$          ;Init "Invalid PW
278A 3E3F      02510      LD      A,63              ;Set extended error
278C B7        02520      OR      A                ;Set NZ condition
278D D1        02530      POP     DE                ;Clean up stack
278E C9        02540      RET
                02550 ;
                02560 ;      Reset any file open bits
                02570 ;
278F E5        02580 UNOPEN PUSH    HL                ;Save buffer posn
2790 C5        02590      PUSH    BC
2791 0608      02600      LD      B,8                ;8 entries
2793 2C        02610      INC     L                ;Dir + 1
2794 CBAE      02620 ZAP     RES      5,(HL)            ;Clear file open bit
2796 3E20      02630      LD      A,32
2798 85        02640      ADD     A,L                ;Pt to next Dir+1
2799 6F        02650      LD      L,A
279A 10F8      02660      DJNZ    ZAP                ;Do 8 entries per direc
279C C1        02670      POP     BC
279D E1        02680      POP     HL
279E C9        02690      RET
                02700 ;
279F          02710 $DSPLY @@DSPLY                ;Display a line
                00014      IFEQ    00H,1
                00015      LD      HL,
                00016      ENDIF
279F 3E0A      00017      LD      A,10
27A1 EF        00018      RST      40
27A2 C8        02720      RET      Z
27A3 181D      02730      JR      IOERR
                02740 ;
27A5          02750 WRSYS  @@WRSSC                ;Write the sector
27A5 3E36      00019      LD      A,54
27A7 EF        00020      RST      40
27A8 2018      02760      JR      NZ,IOERR
27AA          02770      @@VRSEC                ;Verify it
27AA 3E32      00021      LD      A,50
27AC EF        00022      RST      40
27AD FE06      02780      CP      6                ;Must be SYSTEM sector
27AF C8        02790      RET      Z
27B0 1810      02800      JR      IOERR
                02810 ;
27B2          02820 WRSEC  @@WRSEC                ;Write normal sector
27B2 3E35      00023      LD      A,53
27B4 EF        00024      RST      40
27B5 C8        02830      RET      Z
27B6 180A      02840      JR      IOERR

```

```

02850 ;
02860 ;           Sector read routine
02870 ;
27B8 21002A 02880 RDSEC LD      HL,BUF1           ;Read sector
27BB        02890 @RDSEC
27BB 3E31    00025 LD      A,49
27BD EF     00026 RST     40
27BE C8     02900 RET     Z
27BF FE06   02910 CP      6
27C1 C8     02920 RET     Z           ;Fall thru to error?
02930 ;
02940 ;           Error exits
02950 ;
27C2 FE3F   02960 IOERR CP      63           ;Extended error?
27C4 281F   02970 JR      Z,EXTERR           ;Log it and quit
27C6 2600   02980 LD      H,0           ;Error to HL
27C8 6F     02990 LD      L,A
27C9 E5     03000 PUSH    HL           ;Save error code
27CA F6C0   03010 OR      0C0H           ;Set short, return
27CC 4F     03020 LD      C,A           ;Error to C for
27CD        03030 @ERROR           ; display
27CD 3E1A   00027 LD      A,26
27CF EF     00028 RST     40
03040 ;
27D0 21E828 03050 LD      HL,ABTJOB$           ;Init"Job aborted
03060 ;
27D3        03070 @LOGOT           ;Log the msg
00029      IFEQ  00H,1
00030      LD      HL,
00031      ENDIF
27D3 3E0C   00032 LD      A,12
27D5 EF     00033 RST     40
27D6 E1     03080 POP      HL           ;Recover error code
27D7 1812   03090 JR      QUIT$$
03100 ;
03110 ;           Internal error handler
03120 ;
27D9 213429 03130 NIXHARD LD      HL,NIXHARD$           ;"Can't to hard drive
27DC DD     03140 DB      0DDH
27DD 21F728 03150 NOT0 LD      HL,NOT0$           ;"Can't do drive 0
27E0 DD     03160 DB      0DDH
27E1 210C29 03170 PRMERR LD      HL,PRMERR$           ;"Parm error
27E4 DD     03180 DB      0DDH
27E5        03190 EXTERR @LOGOT           ;Display the error
00034      IFEQ  00H,1
00035      LD      HL,
00036      ENDIF
27E5 3E0C   00037 LD      A,12
27E7 EF     00038 RST     40
27E8 21FFFF 03200 LD      HL,-1           ;Set abort code
27EB C31B26 03210 QUIT$$ JP      QUIT$
03220 ;
03230 ;           Read the granule allocation table
03240 ;
27EE F6     03250 GATRD DB      0F6H           ;Set NZ for test
27EF AF     03260 GATWR XOR      A           ;Set Z for test
27F0 E5     03270 PUSH    HL
27F1 F5     03280 PUSH    AF
27F2 FD5609 03290 LD      D,(IY+9)           ;Dir cylinder
27F5 21002A 03300 LD      HL,BUF1
27F8 5D     03310 LD      E,L           ;Set to sector 0

```

```

27F9 F1      03320      POP      AF
27FA 2807    03330      JR        Z,GATRW1      ;Go if write
27FC         03340      @RDSSC
27FC 3E55    00039      LD        A,85
27FE EF      00040      RST       40
27FF 3E14    03350      LD        A,14H
2801 180C    03360      JR        GATRW3
2803         03370  GATRW1 @WRSSC
2803 3E36    00041      LD        A,54
2805 EF      00042      RST       40
2806 2003    03380      JR        NZ,GATRW2      ;Skip verify if error
2808         03390      @VRSEC      ;Verify the write
2808 3E32    00043      LD        A,50
280A EF      00044      RST       40
280B FE06    03400  GATRW2 CP        6      ;Expect error 6
280D 3E15    03410      LD        A,15H      ;Init "Gat error
280F E1      03420  GATRW3 POP      HL
2810 C9      03430      RET
          03440 ;
          03450 ;      Routine to check on floppy present
          03460 ;
2811 3E28    03470  CKDRV LD        A,40      ;@DCSTAT
2813 EF      03480      RST       28H
2814 202F    03490      JR        NZ,ILLEG
2816 3E2C    03500      LD        A,44      ;@RSTORE
2818 EF      03510      RST       28H
2819 21002A  03520      LD        HL,BUF1      ;Set up for
281C C5      03530      PUSH     BC      ; mini ckdrv
281D         03540      @TIME      ;P/u timer ptr
281D 3E13    00045      LD        A,19
281F EF      00046      RST       40
2820 C1      03550      POP      BC
2821 EB      03560      EX        DE,HL      ;Pt HL to
2822 2B      03570      DEC       HL      ; heartbeat counter
2823 3E2F    03580      LD        A,47      ;@RSLCT
2825 EF      03590      RST       28H      ;Wait till ready
2826 7E      03600      LD        A,(HL)      ;Get heartbeat count
2827 C614    03610      ADD       A,20      ;Init to + 500ms
2829 57      03620      LD        D,A      ;Store for timeout check
282A CD3A28  03630  CK1  CALL     INDEX
282D 20FB    03640      JR        NZ,CK1      ;Get no pulse
282F CD3A28  03650  CK2  CALL     INDEX
2832 28FB    03660      JR        Z,CK2      ;Get pulse
2834 CD3A28  03670  CK3  CALL     INDEX
2837 20FB    03680      JR        NZ,CK3      ;Get no pulse
2839 C9      03690      RET
          03700 ;
283A 7E      03710  INDEX LD        A,(HL)      ;Get time
283B BA      03720      CP        D      ;Interval expired?
283C 2806    03730      JR        Z,ILLG1
283E 3E2F    03740      LD        A,47      ;@RSLCT
2840 EF      03750      RST       28H
2841 CB4F    03760      BIT       1,A      ;Test for index pulse
2843 C9      03770      RET
          03780 ;
2844 E1      03790  ILLG1 POP      HL      ;Fix stack
2845 3E20    03800  ILLEG LD        A,32      ;'illegal drv #'
2847 C3C227  03810      JP        IOERR
          03820 ;
          03830 ;
          03840 ;      Messages

```

```

03850 ;
284A 52 03860 HELLO$ DB 'REPAIR'
45 50 41 49 52
2850 03870 *GET CLIENT:3
03950 ;CLIENTS/ASM - File to establish sign-on headers
03960 ;
2850 20 03970 DB ' - 6.2.0 - Copyright 1982/83/84 by Logical'
2D 20 36 2E 32 2E 30 20
2D 20 43 6F 70 79 72 69
67 68 74 20 31 39 38 32
2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61
6C
287A 20 03980 DB ' Systems, Inc. ',10
53 79 73 74 65 6D 73 2C
20 49 6E 63 2E 20 20 20
20 20 20 0A
03990 ;
288F 41 04000 DB 'All Rights Reserved. Licensed 1982/83/84'
6C 6C 20 52 69 67 68 74
73 20 52 65 73 65 72 76
65 64 2E 20 4C 69 63 65
6E 73 65 64 20 31 39 38
32 2F 38 33 2F 38 34
28B7 20 04010 DB ' to xxxxxxxxxxxxxxxxxxxx',10,13
74 6F 20 78 78 78 78 78
78 78 78 78 78 78 78 78
78 78 78 78 78 0A 0D
03880 ;
28CF 52 03890 ALCAO$ DB 'Repair function complete',CR
65 70 61 69 72 20 66 75
6E 63 74 69 6F 6E 20 63
6F 6D 70 6C 65 74 65 0D
28E8 52 03900 ABTJOB$ DB 'REPAIR aborted',CR
45 50 41 49 52 20 61 62
6F 72 74 65 64 0D
28F7 43 03910 NOT0$ DB 'Can''t REPAIR drive 0',CR
61 6E 27 74 20 52 45 50
41 49 52 20 64 72 69 76
65 20 30 0D
290C 50 03920 PRMERR$ DB 'Parameter error',CR
61 72 61 6D 65 74 65 72
20 65 72 72 6F 72 0D
291C 49 03930 BADMPW$ DB 'Invalid master password',CR
6E 76 61 6C 69 64 20 6D
61 73 74 65 72 20 70 61
73 73 77 6F 72 64 0D
2934 43 03940 NIXHARD$ DB 'Can''t repair a hard drive',CR
61 6E 27 74 20 72 65 70
61 69 72 20 61 20 68 61
72 64 20 64 72 69 76 65
0D
03950 ;
294E 80 03960 PRMTBL$ DB 80H
0020 03970 STR EQU 20H
294F 23 03980 DB STR!3,'MPW'
4D 50 57
2953 00 03990 MRSP DB 0
2954 0727 04000 DW MPARM+1
2956 00 04010 NOP
04020 ;

```

0008	04030	PSWDBUF	DS	8	;Password buffer
0004	04040	HASHBUF	DS	4	;Owner & user hashes
0020	04050	FCB	DS	32	
2A00	04060		ORG	\$<-8+1<+8	
0100	04070	BUF1	DS	256	
	04080	;			
2600	04090		END	BEGIN	

\$DSPLY	279F \$EXIT	2618 @@1	0000
@@2	0000 @@3	0000 @@4	0000
@MOD2	0000 @MOD4	FFFF ABB	0010
ABTJOB\$	28E8 ALCAO\$	28CF ALIEN1	26A4
ALIEN2	26AB ALIEN3	26E0 ALIEN4	26FA
BADMPW\$	291C BEGIN	2600 BEG1NA	2609
BLNKMPW	4296 BUF1	2A00 CK1	282A
CK2	282F CK3	2834 CKDRV	2811
CKMPW	2753 CKMPW1	275A CKMPW2	2768
CKMPW3	277A CKMPW4	277C CKMPW5	277F
CKMPW7	2783 CR	000D DRIVE	271C
EXTERR	27E5 FCB	2963 FLAG	0040
GATRD	27EE GATRW1	2803 GATRW2	280B
GATRW3	280F GATWR	27EF GETMPW	272D
GMPW1	2734 GMPW2	273A GMPW4	274E
HASHBUF	295F HELLO\$	284A ILLEG	2845
ILLG1	2844 INDEX	283A INVMPW	2787
IOERR	27C2 LC	26A0 LF	000A
MPARM	2706 MRSP	2953 NIXHARD	27D9
NIXHARD\$	2934 NOT0	27DD NOT0\$	28F7
PGRM	2622 PRMERR	27E1 PRMERR\$	290C
PRMTBL\$	294E PSWDBUF	2957 QUIT\$	261B
QUIT\$	27EB RDSEC	27B8 STACK	261C
STR	0020 UNOPEN	278F WRSEC	27B2
WRSYS	27A5 ZAP	2794 @@ABORT	8EE9
@@ADTSK	8F7C @@BANK	9494 @@BKSP	9174
@@BREAK	94AA @@CHNIO	8ED4 @@CKBRKC	94F8
@@CKDRV	8FD0 @@CKEOF	9189 @@CKTSK	8F67
@@CLOSE	915F @@CLS	94E2 @@CMNDI	8F13
@@CMNDR	8F28 @@CTL	8D38 @@DATE	8EAA
@@DCSTAT	900F @@DEBUG	8F52 @@DECHEX	9414
@@DIRRD	9381 @@DIRWR	9396 @@DIV16	93FF
@@DIV8	93EA @@DODIR	8FE5 @@DSP	8CFC
@@DSPLY	8D9C @@ERROR	8F3D @@EXIT	8EFE
@@FEXT	92EE @@FLAGS	947E @@FNAME	9303
@@FSPEC	92D9 @@GATRD	936C @@GATWR	93AB
@@GET	8D10 @@GTDCB	932D @@GTDCCT	9318
@@GTMOD	9342 @@HDFMT	90B7 @@HEX16	9453
@@HEX8	943E @@HEXDEC	9429 @@HIGH\$	9468
@@INIT	9135 @@KBD	8D74 @@KEY	8CE8
@@KEYIN	8D88 @@KLTSK	8FBB @@LOAD	92AF
@@LOC	919E @@LOF	91B3 @@LOGGER	8DD3
@@LOGOT	8DE8 @@MSG	8E1F @@MUL16	93D5
@@MUL8	93C0 @@OPEN	914A @@PARAM	8E95
@@PAUSE	8E80 @@PEOF	91C8 @@POSN	91DD
@@PRINT	8E34 @@PRT	8D4C @@PUT	8D24
@@RAMDIR	8FFA @@RDSEC	908D @@RDSSC	9357
@@READ	91F2 @@REMOV	9120 @@RENAM	910B
@@REW	9207 @@RMTSK	8F91 @@RPTSK	8FA6
@@RREAD	921C @@RSLCT	9078 @@RSTOR	9039
@@RUN	92C4 @@RWRTIT	9231 @@SEEK	9063
@@SEEKSC	9246 @@SKIP	925B @@SLCT	9024
@@STEPS	904E @@TIME	8EBF @@VDCTL	8E6B
@@VER	9270 @@VRSEC	90A2 @@WEOF	9285
@@WHERE	8D60 @@WRITE	929A @@WRSEC	90CC
@@WRSSC	90E1 @@WRTRK	90F6	

2600 is the transfer address
00000 Total errors

NOTES:

TAPE100/CMD - Read or write a Model 100 tape

Tape100 allows cassette tapes written on a Model 100 to be read in and save as a disk file, and vice versa.


```

00100 ;TAPE100 - Tape/Disk & Disk/Tape Xfer Utility
00110 TITLE <TAPE100 - LS-DOS 6.2>
00120 ;
F440 00130 BREAKLC EQU 0F440H ;<BREAK> key location
003A 00140 LOADA EQU 3AH ; LD A,(nnnn) opcode
0016 00150 WRMASK EQU 'W'-'A' ;WRINTMASK port mask byte
000C 00160 MODMASK EQU 'M'-'A' ;MODOUT port mask byte
00170 ;
003A 00180 @INIT EQU 58 ;@INIT SVC #
003B 00190 @OPEN EQU 59 ;@OPEN SVC #
00200 ;
00E0 00210 PORTE0 EQU 0E0H
00EC 00220 MODOUT EQU 0ECH
00FF 00230 PORTFF EQU 0FFH
0078 00240 OPREG$ EQU 78H ;Operating Register
0084 00250 @OPREG EQU 84H ;Video/Keyboard Control Port
F800 00260 VIDEO EQU 0F800H ;Start of Video RAM
00270 ;
0022 00280 WHICH1 EQU 22H ;Which one - 0 or 1 ?
000F 00290 TOOSHRT EQU 0FH ;Pulse too Short ?
003E 00300 TOOLONG EQU 3EH ;Pulse too Long ?
0006 00310 ROUTOFF EQU 6 ;Interrupt rout offset
000D 00320 DIFFER EQU 0DH ;Difference between 2 pulses
2B2F 00330 DELAY0 EQU 2B2FH ;Bit = 0 Delay count
1217 00340 DELAY1 EQU 1217H ;Bit = 1 Delay count
00350 ;
000E 00360 CURON EQU 14 ;Cursor on
000F 00370 CUROFF EQU 15 ;Cursor off
00380 ;
0000 00390 *GET SVCMAC:3 ;SVC Macro equivalents
00010 ;SVMAC/ASM - LS-DOS Version VI
00020 *LIST OFF
00390 *LIST ON
0000 00400 *GET VALUES:3 ;Misc. equates
00390 ;VALUES/ASM - Version 6
00390 *LIST OFF
00420 *LIST ON
0000 00410 *GET COPYCOM:3 ;Copyright message
00210 ; COPYCOM - File for Copyright COMment block
00420 ;
0000 00430 COM '<*(C) 1982,83,84 by LSI*>'
00420 ;
2600 00430 ORG 2600H
00440 ;
00450 START
2600 00460 @@CKBRKC ;Check for break
2600 3E6A 00001 LD A,106
2602 EF 00002 RST 40
2603 2804 00470 JR Z,STARTA ;Continue if not
2605 21FFFF 00480 LD HL,-1 ; else abort
2608 C9 00490 RET
00500 ;
2609 ED735C27 00510 STARTA LD (OLDSP+1),SP ;Save entry stack
260D CDC727 00520 CALL DOINIT ;Do initialization
00530 ;
00540 ; Was READ or WRITE entered ?
00550 ;
2610 3AD729 00560 LD A,(RRESP) ;P/u read response
2613 47 00570 LD B,A ;Xfer to B
2614 3ACF29 00580 LD A,(WRESP) ;P/u write response
2617 A8 00590 XOR B ;Are both the same ?

```

```

2618 2807      00600           JR        Z,INP_R_W           ;Yes - prompt
              00610 ;
              00620 ;           Both weren't entered - which one was
              00630 ;
261A 04       00640 CHKPRM INC        B                   ;READ entered ?
261B 05       00650           DEC       B
261C 281C     00660           JR        Z,WRTAPE           ;<W>rite a tapefile
261E C3B126   00670           JP        RDTAPE           ;<R>ead a tapefile
              00680 ;
              00690 ;           Prompt for READ or WRITE
              00700 ;
2621 E5       00710 INP_R_W PUSH      HL                   ;Save command ptr
              00720 ;
2622 21DB28   00730           LD        HL,RDORWR          ;"Read or Write"
2625 CD4928   00740           CALL     DSPLY
              00750 ;
              00760 ;           Input R (Read) or W (Write)
              00770 ;
2628 0601     00780           LD        B,1                ;Take input, 1 char
262A CD3528   00790           CALL     INPUT
262D 7E       00800           LD        A,(HL)            ;P/u first char
262E E1       00810           POP       HL                ;Recover command ptr
262F CBAF     00820           RES       5,A               ;Convert to U/C
2631 FE52     00830           CP        'R'               ;<R>ead ?
2633 CAB126   00840           JP        Z,RDTAPE
2636 FE57     00850           CP        'W'               ;<W>rite ?
2638 20E7     00860           JR        NZ,INP_R_W          ;No - re-prompt
              00870 ;
              00880 ;           WRITE diskfile to tapefile
              00890 ;
263A 11F12D   00900 WRTAPE LD        DE,FCB1            ;DE => Source FCB
263D           00910           @@FSPEC                ;If a bad spec,
263D 3E4E     009003           LD        A,78
263F EF       009004           RST       40
2640 C40828   00920           CALL     NZ,PRSOUR           ; prompt for source
              00930 ;
              00940 ;           WRITE - check if destination filespec input
              00950 ;
2643 11112E   00960 WRTAPE2 LD        DE,FCB2            ;DE => Destination FCB
2646           00970           @@FSPEC
2646 3E4E     009005           LD        A,78
2648 EF       009006           RST       40
2649 C41028   00980           CALL     NZ,PRDEST           ;Prompt for destination
264C CDAC27   00990           CALL     GTFILE            ;Xfer into Filename
              01000 ;
              01010 ;           Open Disk Source file
              01020 ;
264F 11F12D   01030 OPDSRC LD        DE,FCB1            ;DE => Source
2652 CDCF2C   01040           CALL     OPEN
2655 C24627   01050           JP        NZ,IOERR            ;NZ - abort
              01060 ;
              01070 ;           Can this disk file fit into memory ?
              01080 ;
2658 2AFD2D   01090           LD        HL,(FCB1+12)        ;P/u ERN
265B 24       01100           INC       H                ;Too big ?
265C 25       01110           DEC       H
265D C2BF29   01120           JP        NZ,TOOBIG            ;Yes - forget it
2660 3E00     01130 ENUF    LD        A,$-$            ;Enough memory ?
2662 C630     01140           ADD       A,MEM<-8           ;Add mem start
2664 BD       01150           CP        L
2665 DABF29   01160           JP        C,TOOBIG            ;No - forget it

```

```

01170 ;
01180 ;      Read in Disk file & Write to tape
01190 ;
2668 CD3B2A 01200 CALL    PRTAPE      ;Display "Ready Tape"
266B CDBC2C 01210 CALL    CURSOFF     ;Turn of cursor
266E 217B27 01220 LD      HL,READING  ;Init "Reading : "
2671 CD4928 01230 CALL    DSPLY      ;Display line
2674 219D27 01240 LD      HL,DFBUF    ;HL => Disk Filename
2677 CD4928 01250 CALL    DSPLY      ;
267A CD692D 01260 CALL    READSRC     ;Read the source file
267D CDCA2D 01270 CALL    GETPOS     ;Get new cursor pos
2680 CD9B2D 01280 CALL    ENDOKI     ;Bring in Video
2683 218527 01290 LD      HL,WRITING  ;Display "Writing : "
2686 CD802C 01300 CALL    DISPSTR    ;
2689 219027 01310 LD      HL,FILENM   ;"filenm"
268C CD802C 01320 CALL    DISPSTR    ;
268F CD1A2A 01330 CALL    CASSON     ;Turn on cassette
2692 0680    01340 LD      B,80H      ;Pause a bit
2694        01350 @@PAUSE
2694 3E10    00007 LD      A,16
2696 EF     00008 RST      40
2697 CDEC2B 01360 CALL    WRHEAD     ;Write Header
269A CD192C 01370 CALL    WRDAT      ;Write Data
269D 2A6627 01380 LD      HL,(CURPOS) ;P/u new cursor pos
26A0 CDE32D 01390 CALL    GETCRS     ;
26A3 0603    01400 LD      B,3       ;Give to system
26A5        01410 @@VDCTL
26A5 3E0F    00009 LD      A,15
26A7 EF     00010 RST      40
26A8 CDA92D 01420 CALL    DISDOKI    ;Restore video
26AB CD2B2A 01430 CALL    CASSOFF    ;Turn off cassette
26AE C35827 01440 JP      EXIT       ;Clean exit
01450 ;
01460 ;      Get Source & Destination for READ
01470 ;
26B1 11F12D 01480 RDTAPE LD      DE,FCB1      ;First filespec legal ?
26B4        01490 @@FSPEC
26B4 3E4E    00011 LD      A,78
26B6 EF     00012 RST      40
26B7 280D    01500 JR      Z,CHKSEC      ;Yes - check for second
01510 ;
01520 ;      Accept first filename on tape
01530 ;
26B9 3EC9    01540 LD      A,0C9H
26BB 320D2B 01550 LD      (CORRECT),A
26BE 11112E 01560 LD      DE,FCB2      ;Prompt for dest filename
26C1 CD2D28 01570 CALL    PRDEST2     ;Prompt for dest
26C4 1825    01580 JR      READFIL    ; and read file
01590 ;
01600 ;      Copy source FCB into destination
01610 ;
26C6 E5     01620 CHKSEC PUSH    HL      ;Save comm ptr
26C7 EB     01630 EX      DE,HL
26C8 11112E 01640 LD      DE,FCB2      ;DE => Disk FCB
26CB 012000 01650 LD      BC,32
26CE D5     01660 PUSH    DE      ;Save dest FCB
26CF EDB0    01670 LDIR
26D1 D1     01680 POP     DE
26D2 E1     01690 POP     HL
01700 ;
01710 ;      P/u destination filespec

```

```

01720 ;
26D3 2B 01730 DEC HL ;Skip leading spaces
26D4 23 01740 SKPSPC INC HL
26D5 7E 01750 LD A,(HL) ;P/u char
26D6 FE20 01760 CP ' ' ;Space ?
26D8 28FA 01770 JR Z,SKPSPC
26DA FE0E 01780 CP CR+1 ;Eol ?
26DC 3807 01790 JR C,GTFILE2 ;Yes - use default
26DE FE28 01800 CP '(' ;Eol ?
26E0 2803 01810 JR Z,GTFILE2
26E2 01820 @@FSPEC ;Xfer in if legal
26E2 3E4E 00013 LD A,78
26E4 EF 00014 RST 40
01830 ;
01840 ; Transfer filename into buffer left just'd
01850 ;
26E5 11F12D 01860 GTFILE2 LD DE,FCB1 ;DE => Source
26E8 CDAC27 01870 CALL GTFILE ;Stuff Filename into buff
01880 ;
01890 ; Read in Tape Source file
01900 ;
26EB 11112E 01910 READFIL LD DE,FCB2 ;@INIT the dest file
26EE CDCB2C 01920 CALL INIT
26F1 C24627 01930 JP NZ,IOERR
26F4 DD4E06 01940 LD C,(IX+6) ;P/u drive #
26F7 01950 @@CKDRV ;Write protected ?
26F7 3E21 00015 LD A,33
26F9 EF 00016 RST 40
26FA 3E0F 01960 LD A,15 ;Write Protected Disk
26FC DA4627 01970 JP C,IOERR ;Good bye
26FF CD3B2A 01980 CALL PRTAPE ;"Ready Cassette"
2702 CD8C2C 01990 CALL CURSOFF
2705 CD9B2D 02000 CALL ENDOKI ;Bring in KI & DO RAM
2708 CDCA2D 02010 CALL GETPOS ;Calculate cursor posn
270B 217B27 02020 LD HL,READING ;Display "Reading : "
270E CD802C 02030 CALL DISPSTR
2711 CD1A2A 02040 CALL CASSON ;Turn on cassette
2714 CDD82A 02050 CALL RDHEAD ;Search for header
2717 CD492A 02060 CALL RDDAT ;Read in Data
271A F3 02070 DI ;Make sure off
271B CD2B2A 02080 CALL CASSOFF ;Turn off cassette
271E 218527 02090 LD HL,WRITING ;Display "Writing : "
2721 CD802C 02100 CALL DISPSTR ;
2724 219D27 02110 LD HL,DFBUF ;HL => Destination
2727 CD802C 02120 CALL DISPSTR ;
272A 2A6627 02130 LD HL,(CURPOS) ;P/u new cursor position
272D CDE32D 02140 CALL GETCRS ;Convert to Row, Column
2730 0603 02150 LD B,3 ;Give system new cursor
2732 02160 @@VDCTL ;
2732 3E0F 00017 LD A,15
2734 EF 00018 RST 40
2735 CDA92D 02170 CALL DISDOKI ;Enable real RAM
2738 1806 02180 JR WRTDES2 ;
273A CDA92D 02190 FORNOW CALL DISDOKI ;Enable real RAM
273D CD2B2A 02200 CALL CASSOFF ;Turn off cassette
2740 CD512D 02210 WRTDES2 CALL WRTDEST ;Write Destination file
2743 C35827 02220 JP EXIT ;Clean exit
02230 ;
02240 ;
2746 6F 02250 IOERR LD L,A ;Xfer error # to HL
2747 2600 02260 LD H,0 ;

```

```

2749 F6C0 02270 OR 0C0H ;Abbrev, return
274B 4F 02280 LD C,A
274C 02290 @@ERROR ;Display error
274C 3E1A 00019 LD A,26
274E EF 00020 RST 40
274F 180A 02300 JR OLDSP ; and abort
02310 ;
2751 C35427 02320 ILLEGAL JP ABORT ;For now
02330 ;
2754 21FFFF 02340 ABORT LD HL,-1 ;Show error return
2757 DD 02350 DB 0DDH ;Skip LD HL,0
2758 210000 02360 EXIT LD HL,0 ;Clean exit
275B 310000 02370 OLDSP LD SP,$-$ ;P/u original SP
275E FB 02380 EI ;Re-enable interrupts
275F 02390 @@CKBRKC ;Clear Break
275F 3E6A 00021 LD A,106
2761 EF 00022 RST 40
2762 C9 02400 RET ; and RETURN
02410 ;
2763 00 02420 DLEN DB 0,0,0
00 00
2766 0000 02430 CURPOS DW 0 ;Cursor Position
2768 0A 02440 READERR DB LF,'Tape Read Error ',CR
54 61 70 65 20 52 65 61
64 20 45 72 72 6F 72 20
20 0D
277B 52 02450 READING DB 'Reading: ',ETX
65 61 64 69 6E 67 3A 20
03
2785 0A 02460 WRITING DB LF,'Writing: ',ETX
57 72 69 74 69 6E 67 3A
20 03
2790 46 02470 FILENM DB 'FILENM',CR
49 4C 45 4E 4D 0D
0006 02480 BUFFER DS 6
279D 46 02490 DFBUF DB 'Filename/ext:d',ETX
69 6C 65 6E 61 6D 65 2F
65 78 74 3A 64 03
02500 ;
02510 ;
02520 ;
02530 ;
02540 ;
27AC 219027 02550 GTFILN LD HL,FILENM ;HL => Filename buffered
27AF E5 02560 PUSH HL ;Save it
27B0 0606 02570 LD B,6 ;Init to all spaces
27B2 3620 02580 CLEAN LD (HL),' '
27B4 23 02590 INC HL
27B5 10FB 02600 DJNZ CLEAN
27B7 E1 02610 POP HL ;HL => Filename dest
27B8 0606 02620 LD B,6 ;Only accept first 6
02630 ;
27BA 1A 02640 GETFILN LD A,(DE) ;P/u char
27BB FE0E 02650 CP CR+1 ;End ?
27BD D8 02660 RET C ;Yes - done
27BE FE2E 02670 CP '.' ;Start of password?
27C0 C8 02680 RET Z ;Yes - done
27C1 77 02690 LD (HL),A ;Stuff into filename buff
27C2 23 02700 INC HL ;Bump
27C3 13 02710 INC DE
27C4 10F4 02720 DJNZ GETFILN

```

```

27C6 C9      02730      RET                      ;Done - RETURN
              02740 ;
              02750 ;      DOINIT - Do initialization
              02760 ;
27C7          02770 DOINIT @@FLAGS                ;IY => System Flags
27C7 3E65     00023      LD      A,101
27C9 EF       00024      RST      40
              02780 ;
              02790 ;      Calculate highest mem address of buffer
              02800 ;
27CA E5       02810      PUSH     HL                ;Save command line stuff
27CB 210000    02820      LD      HL,0              ;P/u HIGH$
27CE 45        02830      LD      B,L
27CF FDCB024E 02840      BIT      1,(IY+CFLAG$)    ;@CMNDR ?
27D3 2801      02850      JR      Z,USEHI
27D5 04        02860      INC      B                ;Use LOW$
27D6          02870 USEHI @@HIGH$
27D6 3E64     00025      LD      A,100
27D8 EF       00026      RST      40
27D9 23        02880      INC      HL                ;Set hi-mem byte
27DA 25        02890      DEC      H                ;Give some lee-way
27DB 25        02900      DEC      H
27DC 7C        02910      LD      A,H                ; & stuff in R/W routines
27DD 326126    02920      LD      (ENUF+1),A
              02930 ;
              02940 ;      Display Log-on message
              02950 ;
27E0 215328    02960      LD      HL,HELLO$          ;Display banner
27E3 CD4928    02970      CALL    DSPLY
27E6 E1        02980      POP      HL                ;Process parm line
              02990 ;
              03000 ;      P/u READ or WRITE parm if entered
              03010 ;
27E7 E5       03020      PUSH     HL                ;Save HL
27E8 2B        03030      DEC      HL                ;Back up one
27E9 23        03040 CKPLP INC      HL                ;Bump
27EA 7E        03050      LD      A,(HL)            ;P/u char
27EB FE0E      03060      CP      CR+1              ;Eol ?
27ED 380D      03070      JR      C,DUNLIN          ;Yes - done
27EF FE28      03080      CP      '('              ;Paramter entered ?
27F1 20F6      03090      JR      NZ,CKPLP          ;No - go til eol
              03100 ;
              03110 ;      Process parameter entry
              03120 ;
27F3 11C829    03130      LD      DE,PARMTBL        ;DE => Param table
27F6          03140      @@PARAM
27F6 3E11     00027      LD      A,17
27F8 EF       00028      RST      40
27F9 C2BB29    03150      JP      NZ,PRMERR          ;NZ - parameter error
27FC E1        03160 DUNLIN POP      HL            ;Rcvr command ptr
              03170 ;
              03180 ;      If C=N entered then use checksum
              03190 ;
27FD 01FFFF    03200 CPARM LD      BC,0FFFFH        ;Default no checksum
2800 04        03210      INC      B                ;User requesting checksum?
2801 C0        03220      RET      NZ                ;Yes, return
2802 3EC9      03230      LD      A,0C9H            ;Init RET opcode
2804 32242B    03240      LD      (CHKERR+1),A      ;Stuff into Checksum error
2807 C9        03250      RET
              03260 ;
              03270 ;      PRSOUR/PRDEST - Prompt for Source & Destination

```

```

03280 ;
2808 E5 03290 PRSOUR PUSH HL ;Save HL
2809 210A29 03300 LD HL,DSF ;"Disk Source Filename"
280C 0617 03310 LD B,23 ;23 chars max
280E 1806 03320 JR DOINPUT
2810 E5 03330 PRDEST PUSH HL ;Save HL
2811 212329 03340 LD HL,TDF ;"Tape Dest Filename"
2814 0606 03350 LD B,6 ;6 char max
2816 CD4928 03360 DOINPUT CALL DSPLY ;Display prompt
2819 E5 03370 PUSH HL ;Save prompt start
281A CD3528 03380 CALL INPUT ;Input
281D 03390 @@FSPEC ;Legal ?
281D 3E4E 00029 LD A,78
281F EF 00030 RST 40
2820 E1 03400 POP HL ;HL => Prompt string
2821 20F3 03410 JR NZ,DOINPUT ;Reprompt on bad name
2823 E1 03420 POP HL ;Recover ptr
2824 C9 03430 RET ; and return
03440 ;
03450 ; PRSOUR2/PRDEST2 - Prompt for READ source/dest
03460 ;
2825 E5 03470 PRSOUR2 PUSH HL ;Save HL
2826 21F128 03480 LD HL,TSF ;"Tape Source filename"
2829 0606 03490 LD B,6 ;6 char max
282B 18E9 03500 JR DOINPUT
282D E5 03510 PRDEST2 PUSH HL ;Save HL
282E 214129 03520 LD HL,DDF ;"Disk Destination file"
2831 0617 03530 LD B,23 ;23 char max
2833 18E1 03540 JR DOINPUT
03550 ;
03560 ; INPUT - Line input routine
03570 ;
2835 D5 03580 INPUT PUSH DE ;Save DE
2836 C5 03590 PUSH BC ; and BC
2837 21312E 03600 LD HL,INBUFF ;HL => Input buffer
283A 03610 @@KEYIN ;Input line
283A 3E09 00031 LD A,9
283C EF 00032 RST 40
283D DA5427 03620 JP C,ABORT ;<BREAK> abort
2840 C1 03630 POP BC ;Restore regs
2841 D1 03640 POP DE
2842 C9 03650 RET
03660 ;
2843 D5 03670 DSP PUSH DE ;Save DE
2844 03680 @@DSP ;Output char
2844 3E02 00033 LD A,2
2846 EF 00034 RST 40
2847 1804 03690 JR EXDSP
03700 ;
2849 D5 03710 DSPLY PUSH DE ;Save DE
284A 03720 @@DSPLY ;Display message
00035 IFEQ 00H,1
00036 LD HL,
00037 ENDIF
284A 3E0A 00038 LD A,10
284C EF 00039 RST 40
284D D1 03730 EXDSP POP DE ;Rcvr DE
284E C8 03740 RET Z ;RETurn if OK
284F C34627 03750 JP IOERR ; else abort
03760 ;
2852 00 03770 COUNT DB 0 ;Count

```

```

03780 ;
2853 1C 03790 HELLO$ DB 1CH,1FH,'TAPE100'
1F 54 41 50 45 31 30 30
285C 03800 *GET CLIENT:3
04240 ;CLIENTS/ASM - File to establish sign-on headers
04250 ;
285C 20 04260 DB ' - 6.2.0 - Copyright 1982/83/84 by Logical'
2D 20 36 2E 32 2E 30 20
2D 20 43 6F 70 79 72 69
67 68 74 20 31 39 38 32
2F 38 33 2F 38 34 20 62
79 20 4C 6F 67 69 63 61
6C
2886 20 04270 DB ' Systems, Inc. ',10
53 79 73 74 65 6D 73 2C
20 49 6E 63 2E 20 20 20
20 20 20 0A
04280 ;
289B 41 04290 DB 'All Rights Reserved. Licensed 1982/83/84'
6C 6C 20 52 69 67 68 74
73 20 52 65 73 65 72 76
65 64 2E 20 4C 69 63 65
6E 73 65 64 20 31 39 38
32 2F 38 33 2F 38 34
28C3 20 04300 DB ' to xxxxxxxxxxxxxxxxxxxx',10,13
74 6F 20 78 78 78 78 78
78 78 78 78 78 78 78 78
78 78 78 78 78 0A 0D
03810 ;
28DB 3C 03820 RDORWR DB '<R>ead or <W>rite ? ',CURON,ETX
52 3E 65 61 64 20 6F 72
20 3C 57 3E 72 69 74 65
20 3F 20 0E 03
28F1 54 03830 TSF DB 'Tape Source Filespec ? ',CURON,ETX
61 70 65 20 53 6F 75 72
63 65 20 46 69 6C 65 73
70 65 63 20 3F 20 0E 03
290A 44 03840 DSF DB 'Disk Source Filespec ? ',CURON,ETX
69 73 6B 20 53 6F 75 72
63 65 20 46 69 6C 65 73
70 65 63 20 3F 20 0E 03
2923 54 03850 TDF DB 'Tape Destination Filespec ? ',CURON,ETX
61 70 65 20 44 65 73 74
69 6E 61 74 69 6F 6E 20
46 69 6C 65 73 70 65 63
20 3F 20 0E 03
2941 44 03860 DDF DB 'Disk Destination Filespec ? ',CURON,ETX
69 73 6B 20 44 65 73 74
69 6E 61 74 69 6F 6E 20
46 69 6C 65 73 70 65 63
20 3F 20 0E 03
295F 52 03870 TREADY DB 'Ready Cassette & Press <ENTER>'
65 61 64 79 20 43 61 73
73 65 74 74 65 20 26 20
50 72 65 73 73 20 3C 45
4E 54 45 52 3E
297D 0E 03880 DB CURON,ETX
03
297F 50 03890 PRMERR$ DB 'Parameter error',LF,CR
61 72 61 6D 65 74 65 72
20 65 72 72 6F 72 0A 0D

```



```

2990 46            03900 TOOBIG$ DB            'File too large to fit in available '
         69 6C 65 20 74 6F 6F 20
         6C 61 72 67 65 20 74 6F
         20 66 69 74 20 69 6E 20
         61 76 61 69 6C 61 62 6C
         65 20
29B3 6D            03910            DB            'memory',LF,CR
         65 6D 6F 72 79 0A 0D
                 03920 ;
                 03930 ;
                 03940 ;            Error Exit routine
                 03950 ;
29BB 217F29       03960 PRMERR LD           HL,PRMERR$            ;"Parameter Error"
29BE DD           03970            DB           0DDH            ;Skip
29BF 219029       03980 TOOBIG LD           HL,TOOBIG$           ;"File too Big"
                 03990 ;
29C2            04000            @@LOGOT            ;Display error
                 00040            IFEQ       00H,1
                 00041            LD           HL,
                 00042            ENDIF
29C2 3E0C         00043            LD           A,12
29C4 EF           00044            RST        40
29C5 C35427       04010            JP           ABORT            ;Good bye
                 04020 ;
                 04030 ;            Parameter Table
                 04040 ;
29C8 80           04050 PARMTBL DB           80H            ;6.x @PARAM
29C9 55           04060            DB           FLAG!ABB!5
29CA 57           04070            DB           'WRITE'
                 52 49 54 45
29CF 00           04080 WRESP    DB           0
29D0 E629         04090            DW           WPARM
                 04100 ;
29D2 54           04110            DB           FLAG!ABB!4
29D3 52           04120            DB           'READ'
                 45 41 44
29D7 00           04130 RRESP    DB           0
29D8 E429         04140            DW           RPARM
                 04150 ;
29DA 54           04160            DB           FLAG!ABB!4
29DB 43           04170            DB           'CHECK'
                 48 45 43 4B
29E0 00           04180 CRESP    DB           0
29E1 FE27         04190            DW           CPARM+1
                 04200 ;
29E3 00           04210            DB           0
                 04220 ;
29E4 0000         04230 RPARM    DW           0
29E6 0000         04240 WPARM    DW           0
                 04250 ;
29E8 00           04260            DC           50,0            ;Patch space
                 00 00 00 00 00 00 00 00
                 00 00 00 00 00 00 00 00
                 00 00 00 00 00 00 00 00
                 00 00 00 00 00 00 00 00
                 00 00 00 00 00 00 00 00
                 00
                 04270 ;
2A1A            04280 *GET        TAPE100A:3
                 04310 ;TAPE100A/ASM - Tape I/O routines

```

```

04320 ;      CASSON - Turn Cassette Motor On
04330 ;
2A1A F3      04340 CASSON DI      ;Disable interrupts
2A1B CDB52D  04350 CALL      SWAP38      ;Grab RST 38H vector
2A1E DBE0    04360 IN        A,(PORTE0) ;Clear any latches
2A20 DBEC    04370 IN        A,(MODOUT) ;Clear any latches
2A22 3E02    04380 LD        A,2      ;Motor on, slow speed
2A24 D3EC    04390 OUT      (MODOUT),A ;Turn on motor
2A26 3E03    04400 LD        A,3      ;Disable other interrupts
2A28 D3E0    04410 OUT      (PORTE0),A
2A2A C9      04420 RET
04430 ;
04440 ;      CASSOFF - Turn off Cassette Motor
04450 ;
2A2B FD7E16  04460 CASSOFF LD      A,(IY+WRMASK) ;P/u original
2A2E D3E0    04470 OUT      (PORTE0),A ;Set up R/F interrupt
2A30 DBFF    04480 IN        A,(PORTFF) ;Clear 1500 bd interrupts
2A32 FD7E0C  04490 LD        A,(IY+MODMASK) ;Turn off motor
2A35 D3EC    04500 OUT      (MODOUT),A
2A37 CDB52D  04510 CALL      SWAP38      ;Restore RST 38H vector
2A3A C9      04520 RET
04530 ;
04540 ;      PRTAPE - Prompt for "Tape Ready" & turn motor on
04550 ;
2A3B 215F29  04560 PRTAPE LD      HL,TREADY ;"Ready cassette & <ENTER>
2A3E CD4928  04570 CALL      DSPLY
2A41 0601    04580 NOTENT LD      B,1      ;Just 1 char
2A43 CD3528  04590 CALL      INPUT      ;<BREAK> or <ENTER>
2A46 C3BC2C  04600 JP        CURSOFF      ;Turn off Cursor & RETURN
04610 ;
04620 ;      RDDAT - Read in a tape file
04630 ;
2A49 21002F  04640 RDDAT  LD      HL,MEM-100H ;HL => Start of file
2A4C 24      04650 RDDAT2 INC      H      ;Bump hi-byte
2A4D CD582A  04660 CALL      RDDATA      ;Read a block
2A50 C8      04670 RET        Z      ;Eof ?
2A51 3E00    04680 EOTF  LD      A,$-$      ;At top of memory ?
2A53 BC      04690 CP        H      ;No
2A54 20F6    04700 JR        NZ,RDDAT2      ;Top of mem -
2A56 B7      04710 OR        A      ;RETURN NZ
2A57 C9      04720 RET
04730 ;
04740 ;      RDDATA - Read in a block of Data
04750 ;      HL => Destination of Block
04760 ;
2A58 CD392B  04770 RDDATA CALL      RDSYNC      ;Read sync field
2A5B CDA72B  04780 CALL      RDBYTE      ;Read a byte
2A5E FE8D    04790 CP        8DH      ;Legal ?
2A60 C25127  04800 JP        NZ,ILLEGAL      ;No - bad news
2A63 110000  04810 LD        DE,0      ;D=EOF flag, E = checksum
04820 ;
2A66 CDA72B  04830 RDLPI CALL      RDBYTE      ;Read a byte
2A69 77      04840 LD        (HL),A      ;Stuff into buffer
04850 ;
04860 ;      Check for End of File byte X'1A'
04870 ;
2A6A FE1A    04880 CP        1AH      ;Eof ?
2A6C 2005    04890 JR        NZ,AFTER      ;No
2A6E BA      04900 CP        D      ;Been here before ?
2A6F 2802    04910 JR        Z,AFTER      ;First time ?
2A71 57      04920 LD        D,A      ;Set D = 1AH

```

```

2A72 45      04930      LD      B,L          ;Yes - set B = pos
              04940 ;
              04950 ;      Add byte to checksum
              04960 ;

2A73 83      04970 AFTER ADD      A,E          ;Add checksum
2A74 5F      04980      LD      E,A          ;Xfer back to E
2A75 2C      04990      INC     L            ;Bump
2A76 20EE    05000      JR      NZ,RDLP1
2A78 ED44    05010      NEG     A            ;Negate checksum
2A7A 5F      05020      LD      E,A          ;Stuff back in E
              05030 ;
              05040 ;      Verify Checksum byte
              05050 ;

2A7B CDA72B  05060      CALL    RDBYTE        ;Read in byte
2A7E BB      05070      CP      E            ;Checksums match ?
2A7F C4232B  05080      CALL    NZ,CHKERR      ;No - checksum error
              05090 ;
              05100 ;      Stuff EOF offset byte into WRTDEST routine
              05110 ;

2A82 7C      05120      LD      A,H          ;P/u eom
2A83 325C2D  05130      LD      (EOTF2+1),A  ;Stuff into WRTDEST
2A86 78      05140      LD      A,B          ;P/u byte
2A87 3C      05150      INC     A            ;Bump
2A88 32612D  05160      LD      (OFFSET+1),A
              05170 ;
              05180 ;      Read past 20 dummy zeroes
              05190 ;

2A8B 0614    05200      LD      B,20
2A8D CDA72B  05210 RDLP2 CALL    RDBYTE
2A90 10FB    05220      DJNZ    RDLP2
              05230 ;
              05240 ;      Set Z flag if at EOF
              05250 ;

2A92 7A      05260      LD      A,D          ;Eof ?
2A93 FE1A    05270      CP      1AH
2A95 C9      05280      RET
              05290 ;
              05300 ;      RDBIT - Read a Bit from Cassette
              05310 ;

2A96 0E00    05320 RDBIT LD      C,0          ;Init count = 0
2A98 FB      05330      EI                ;Back on
2A99 0C      05340 RBLP  INC     C            ;Bump count
2A9A 3A40F4  05350      LD      A,(BREAKLC)  ;<BREAK> hit ?
2A9D E604    05360      AND     4
2A9F 28F8    05370      JR      Z,RBLP        ;No - wait for interrupt
              05380 ;
              05390 ;      <BREAK> key hit - Abort
              05400 ;

2AA1 F3      05410      DI                ;Cancel next interrupt
2AA2 CDA92D  05420      CALL    DISDOKI      ;Put *DO & *KI back
2AA5 CD2B2A  05430      CALL    CASSOFF      ;Turn off cassette
2AA8 0E0D    05440      LD      C,CR        ;End line
2AAA CD4328  05450      CALL    DSP
2AAD C35427  05460      JP      ABORT        ;Go to abort routine
              05470 ;
              05480 ;      Interrupt Handler - Comes from RST 38
              05490 ;

2AB0 C3B32A  05500 RST38V JP      $+3        ;Wait
2AB3 F5      05510      PUSH    AF          ;Save status
2AB4 DBE0    05520      IN      A,(PORTE0)    ;Read port
2AB6 1F      05530      RRA                ;Bit 0 low ?

```

```

2AB7 D2C12A 05540 JP NC,BIT0LOW
2ABA 1F 05550 RRA ;Bit 1 low ?
2ABB D2C52A 05560 JP NC,BIT1LOW
2ABE F1 05570 POP AF ;Recover status
2ABF FB 05580 EI ;Back on
2AC0 C9 05590 RET ;RETurn
05600 ;
05610 ; Set E = bit image - bit 0 or 1
05620 ;
2AC1 1E01 05630 BIT0LOW LD E,1 ;High
2AC3 1802 05640 JR BIT1LOW+2 ;Add interrupt offset
2AC5 1E00 05650 BIT1LOW LD E,0 ;Low
2AC7 3E06 05660 LD A,ROUTOFF ;Add interrupt routine
2AC9 81 05670 ADD A,C ;Offset to C
2ACA 4F 05680 LD C,A
05690 ;
05700 ; Is the Head on a valid pulse ?
05710 ;
2ACB DBFF 05720 IN A,(PORTFF) ;Read cassette level
2ACD E601 05730 AND 1 ;Mask off all but bit 0
2ACF BE 05740 CP E ;Same as given level ?
2AD0 2003 05750 JR NZ,WAITINT ;No - wait for next inter
05760 ;
05770 ; Valid pulse - Get out of interrupt routine
05780 ;
2AD2 F1 05790 POP AF ;Remove RST 38 RET addr
2AD3 F1 05800 POP AF
2AD4 C9 05810 RET
05820 ;
05830 ; Not the right interrupt - wait for next
05840 ;
2AD5 F1 05850 WAITINT POP AF ;Recover status
2AD6 FB 05860 EI ; and wait for next
2AD7 C9 05870 RET ; interrupt
05880 ;
05890 ; RDHEAD - Read a TAPE100 header
05900 ;
2AD8 2A6627 05910 RDHEAD LD HL,(CURPOS) ;P/u cursor position
2ADB 119727 05920 LD DE,BUFFER ;Buffer
2ADE CD392B 05930 CALL RDSYNC ;Read in SYNC
05940 ;
05950 ; Read in Header Type byte
05960 ;
2AE1 CDA72B 05970 CALL RDBYTE ;Read type byte
2AE4 FE9C 05980 CP 9CH ;Text type ?
2AE6 20F0 05990 JR NZ,RDHEAD ;No - try again
06000 ;
2AE8 010006 06010 LD BC,600H ;B=6 bytes, Checksum = 0
06020 ;
2AEB CDA22B 06030 RFNLP CALL RDBYTEC ;Read byte
2AEE 77 06040 LD (HL),A ;Save byte
2AEF 12 06050 LD (DE),A ;Stuff in buffer
2AF0 23 06060 INC HL ;Bump cursor pos
2AF1 13 06070 INC DE ;Bump buffer ptr
2AF2 10F7 06080 DJNZ RFNLP
06090 ;
06100 ; Next ten bytes are unused
06110 ;
2AF4 060A 06120 LD B,10
2AF6 CDA22B 06130 BOGUSLP CALL RDBYTEC ;Read byte & checksum
2AF9 10FB 06140 DJNZ BOGUSLP

```

```

06150 ;
06160 ;      Negate checksum
06170 ;
2AFB 79      06180 LD      A,C      ;P/u checksum
2AFC ED44    06190 NEG      ;Negate it
2AFE 4F      06200 LD      C,A
2AFF CDA72B  06210 CALL    RDBYTE      ;Read in Checksum byte
2B02 B9      06220 CP      C      ;Match ?
2B03 C4232B  06230 CALL    NZ,CHKERR ;No - checksum error
06240 ;
06250 ;      Read in twenty zeros
06260 ;
2B06 0614    06270 LD      B,20
2B08 CDA72B  06280 DUMBYT CALL    RDBYTE
2B0B 10FB    06290 DJNZ    DUMBYT
06300 ;
06310 ;      Check if this is the correct filename
06320 ;
2B0D 00      06330 CORRECT NOP      ;X'C9' if first filename
2B0E 119727  06340 LD      DE,BUFFER ;Is this the one ?
2B11 219027  06350 LD      HL,FILENM
2B14 0606    06360 LD      B,6      ;6 chars in filename
06370 ;
06380 ;      Loop to compare (HL) to (DE)
06390 ;
2B16 1A      06400 CKFILE LD      A,(DE)      ;P/u header byte
2B17 CDB32C  06410 CALL    CONV_UC      ;Convert to U/C
2B1A BE      06420 CP      (HL)      ;Match ?
2B1B 23      06430 INC      HL
2B1C 13      06440 INC      DE
2B1D C2D82A  06450 JP      NZ,RDHEAD      ;No - try again
2B20 10F4    06460 DJNZ    CKFILE
2B22 C9      06470 RET      ;Yes - RETurn
06480 ;
06490 ;      Checksum error - Either ignore it or "C"
06500 ;
2B23 00      06510 CHKERR NOP      ;RETurn or NOP
2B24 F3      06520 DI      ;Disable interrupts
2B25 3E43    06530 LD      A,'C'      ;<C>checksum error
2B27 324FF8  06540 CHKERR2 LD      (VIDEO+79),A
2B2A CDA92D  06550 CALL    DISDOKI      ;Bring back RAM
2B2D CD2B2A  06560 CALL    CASSOFF      ;Turn off motor
2B30 216827  06570 LD      HL,READERR ;"Tape Read Error!"
2B33 CD4928  06580 CALL    DSPLY
2B36 C35427  06590 JP      ABORT      ;Good bye
06600 ;
06610 ;      RDSYNC - Read Cassette SYNC byte field
06620 ;
06630 ;      Save Registers
06640 ;
2B39 E5      06650 RDSYNC PUSH    HL      ;Save regs
2B3A D5      06660 PUSH    DE
2B3B C5      06670 PUSH    BC
2B3C 3E01    06680 LD      A,1      ;Set interrupt vector
2B3E D3E0    06690 OUT     (PORTE0),A
06700 ;
06710 ;      Read in 128 bits (16 bytes) initially
06720 ;
2B40 0680    06730 RDSYNC2 LD      B,80H      ;Read 128 bits (16 bytes)
2B42 CD962A  06740 RBTLP  CALL    RDBIT      ;Read bit
2B45 79      06750 LD      A,C      ;P/u count value

```

The Source	UTILITY Files	TAPE100 - LS-DOS 6.2	Page 00014
2B46 FE0F	06760	CP TOOSHRT	;Is this a bit ?
2B48 38F6	06770	JR C,RDSYNC2	;No - didn't find a bit
2B4A FE3E	06780	CP TOOLONG	;Is this a bit ?
2B4C 30F2	06790	JR NC,RDSYNC2	;No - wait for bit
2B4E 10F2	06800	DJNZ RBTLP	;Legal bit - dec count
	06810 ;		
	06820 ;		
	06830 ;	Now check parity of next 128 bits	
2B50 210000	06840 RESCNT	LD HL,0	;H = 0's count, L = 1's
2B53 0640	06850	LD B,40H	
	06860 ;		
	06870 ;	Read in 3 bits	
	06880 ;		
2B55 CD962A	06890 LOOP	CALL RDBIT	;Read bit
2B58 CD962A	06900	CALL RDBIT	;Read bit
2B5B 51	06910	LD D,C	;Save count
2B5C CD962A	06920	CALL RDBIT	;Read bit
	06930 ;		
	06940 ;	Calculate Difference between last 2 bits	
	06950 ;		
2B5F 7A	06960	LD A,D	;P/u last bit
2B60 91	06970	SUB C	;Subtract current bit
2B61 3002	06980	JR NC,ABSVAL	
2B63 ED44	06990	NEG	;Change to ABS value
	07000 ;		
	07010 ;	If Value < DIFFER then Bit = 1, else Bit = 0	
	07020 ;		
2B65 FE0D	07030 ABSVAL	CP DIFFER	;Bit = 1 ?
2B67 3803	07040	JR C,BIT1	;Yes - bump Bit 1 count
2B69 24	07050	INC H	;No - bump Bit 0 count
2B6A 1801	07060	JR DODJ	;Back to loop
2B6C 2C	07070 BIT1	INC L	;Bump Bit 1 count
2B6D 10E6	07080 DODJ	DJNZ LOOP	;Dec count - go to loop
	07090 ;		
	07100 ;	Check if H (0's count) & L (1's count) = 40	
	07110 ;		
2B6F 3E40	07120	LD A,40H	;Is H = 64 ?
2B71 BC	07130	CP H	
2B72 280A	07140	JR Z,CHKMARK	;Yes - check for marker
2B74 BD	07150	CP L	;Is L = 64 ?
2B75 20D9	07160	JR NZ,RESCNT	;No - Reset count
	07170 ;		
	07180 ;	Set interrupt Vector & discard 1 bit	
	07190 ;		
2B77 3E02	07200	LD A,2	;Set interrupt vector
2B79 D3E0	07210	OUT (PORTE0),A	
2B7B CD962A	07220	CALL RDBIT	;Read bit
	07230 ;		
	07240 ;	Rotate each bit read in D & check if = X'7F'	
	07250 ;		
2B7E 1600	07260 CHKMARK	LD D,0	;Set byte = 0
2B80 CD962A	07270 GETBIT	CALL RDBIT	;Read next bit
2B83 CD8F2B	07280	CALL ROTBYTE	;Rotate into Byte (D)
2B86 7A	07290	LD A,D	;P/u byte
2B87 FE7F	07300	CP 7FH	;Marker byte ?
2B89 20F5	07310	JR NZ,GETBIT	;No - get another bit
	07320 ;		
	07330 ;	Found marker byte - Restore Regs & RETURN	
	07340 ;		
2B8B C1	07350	POP BC	;Restore Registers
2B8C D1	07360	POP DE	

```

2B8D E1      07370      POP      HL
2B8E C9      07380      RET              ;Done
           07390      ;
           07400      ;      ROTBYTE - Rotate bit through D & check if error
           07410      ;
2B8F 79      07420      ROTBYTE LD      A,C              ;P/u count
2B90 FE22    07430      CP          WHICH1             ;Bit = 0 or 1 ?
2B92 CB12    07440      RL          D                  ;Set bit if Carry set
2B94 FE0F    07450      CP          TOOSHRT            ;Too quick ?
2B96 DA9C2B  07460      JP          C,CIOERR           ;Yes - I/O Error
2B99 FE3E    07470      CP          TOOLONG            ;Too long
2B9B D8      07480      RET          C                  ;No - RETURN
           07490      ;
           07500      ;      Cassette I/O Error - Display Error
           07510      ;
2B9C F3      07520      CIOERR DI          ;Interrupts off
2B9D 3E44    07530      LD          A,'D'              ;Data Error
2B9F C3272B  07540      JP          CHKERR2
           07550      ;
           07560      ;      RDBYTEC - Read byte & Add byte to Check Sum
           07570      ;
2BA2 CDA72B  07580      RDBYTEC CALL     RDBYTE          ;Read byte
2BA5 81      07590      ADD          A,C              ;Add to checksum
2BA6 C9      07600      RET              ;Done
           07610      ;
           07620      ;      RDBYTE - Read a byte
           07630      ;      A <= Byte
           07640      ;
2BA7 D5      07650      RDBYTE: PUSH     DE              ;Save regs
2BA8 C5      07660      PUSH     BC
2BA9 CD962A  07670      CALL     RDBIT              ;Get bogus bit
2BAC 1600    07680      LD          D,0              ;Init byte = 0
2BAE 0608    07690      LD          B,8              ;8 bits to read
           07700      ;
2BB0 CD962A  07710      RDBLP   CALL     RDBIT              ;Read a bit
2BB3 CD8F2B  07720      CALL     ROTBYTE            ;Rotate into D
2BB6 10F8    07730      DJNZ     RDBLP
           07740      ;
           07750      ;      Add to Byte count
           07760      ;
2BB8 3A5228  07770      LD          A,(COUNT)         ;P/u count
2BBB 3C      07780      INC          A                  ; & inc it
2BBC E63F    07790      AND          3FH              ;Ck if the 64th
2BBE 325228  07800      LD          (COUNT),A         ;Save the count
2BC1 2008    07810      JR          NZ,NOTBLNK
           07820      ;
2BC3 3A4FF8  07830      LD          A,(VIDEO+79)        ;Blink every 64
2BC6 EE0A    07840      XOR          0AH
2BC8 324FF8  07850      LD          (VIDEO+79),A
           07860      ;
2BCB 7A      07870      NOTBLNK LD      A,D              ;Xfer byte to A
2BCC 1800    07880      JR          NEXTINS            ;Timing
           07890      ;
2BCE C1      07900      NEXTINS POP      BC              ;Restore BC & DE
2BCF D1      07910      POP      DE
2BD0 C9      07920      RET              ;Done
           07930      ;
           07940      ;      WRBIT - Write a bit to Cassette
           07950      ;
           07960      ;      Set DE = Delay Count for bit
           07970      ;

```

```

2BD1 CB01      07980 WRBIT   RLC      C           ;Get bit
2BD3 3005      07990        JR      NC,NOPULS    ;NC - bit 0
2BD5 111712    08000 BTI     LD      DE,DELAY1    ;Delay for bit 1
2BD8 1803      08010        JR      DEL_LP       ;Go to delay
2BDA 112F2B    08020 NOPULS LD      DE,DELAY0    ;Delay for bit=0
                08030 ;
                08040 ;      Delay 18 counts for 1, 43 counts for 0
                08050 ;
2BDD 15        08060 DEL_LP DEC      D           ;Dec count
2BDE 20FD      08070        JR      NZ,DEL_LP
2BE0 3E02      08080        LD      A,2           ;0 Volts to tape
2BE2 D3FF      08090        OUT     (PORTFF),A
2BE4 1D        08100 DEL_LP2 DEC     E           ;Secondary delay
2BE5 20FD      08110        JR      NZ,DEL_LP2
2BE7 3E01      08120        LD      A,1           ;0.85 volts to tape
2BE9 D3FF      08130        OUT     (PORTFF),A
2BEB C9        08140        RET
                08150 ;
                08160 ;      WRHEAD - Write a cassette header
                08170 ;
2BEC CD602C    08180 WRHEAD CALL    WRSYNC        ;Write SYNC pattern
                08190 ;
                08200 ;      Write Text header type byte X'9C'
                08210 ;
2BEF 1600      08220        LD      D,0           ;Init checksum = 0
2BF1 0E9C      08230        LD      C,9CH        ;Text header type byte
2BF3 CD512C    08240        CALL    WRBYTE       ;Write type byte
                08250 ;
                08260 ;      Write Filename in header block
                08270 ;
2BF6 0606      08280        LD      B,6           ;B = 6 chars
2BF8 219027    08290        LD      HL,FILENM    ;HL => Filename
2BFB 4E        08300 FILELP LD      C,(HL)      ;P/u filename character
2BFC CD4A2C    08310        CALL    WRBYTEC      ; and write it
2BFF 23        08320        INC      HL          ;Bump count
2C00 10F9      08330        DJNZ    FILELP
                08340 ;
                08350 ;      Write 10 filler bytes
                08360 ;
2C02 060A      08370        LD      B,10
2C04 CD4A2C    08380 BOGUS  CALL    WRBYTEC
2C07 10FB      08390        DJNZ    BOGUS
                08400 ;
                08410 ;      Write checksum byte & 20 dummy X'00' bytes
                08420 ;
2C09 7A        08430        LD      A,D           ;P/u checksum
2C0A ED44      08440        NEG
2C0C 4F        08450        LD      C,A           ; & xfer to C
2C0D CD512C    08460        CALL    WRBYTE       ;Write Checksum byte
2C10 010014    08470        LD      BC,1400H     ;B = 20 bytes, C = 0
2C13 CD512C    08480 DUMMY  CALL    WRBYTE      ;Write byte
2C16 10FB      08490        DJNZ    DUMMY
2C18 C9        08500        RET                 ;Get back quick
                08510 ;
                08520 ;      WRDAT - Write a chunk of data to cassette
                08530 ;
2C19 210030    08540 WRDAT  LD      HL,MEM        ;HL => Mem start
2C1C CD272C    08550 WRDAT2 CALL    WRDATA      ;Write Block
2C1F 24        08560        INC      H
2C20 3AF52D    08570        LD      A,(FCB1+4)    ;Finished ?
2C23 BC        08580        CP      H

```



```

2C24 20F6            08590            JR            NZ,WRDAT2            ;No - write another
2C26 C9             08600            RET                            ;Yes - RETURN
                     08610 ;
                     08620 ;            WRDATA - Write a data Block
                     08630 ;            HL => 256 byte block of data (page boundary)
                     08640 ;
2C27 CD602C        08650 WRDATA CALL            WRSYNC            ;Write sync pattern
2C2A 0E8D           08660            LD            C,8DH            ;Write X'8D' type byte
2C2C CD512C        08670            CALL            WRBYTE
                     08680 ;
                     08690 ;            Write 256 byte block of data
                     08700 ;
2C2F AF             08710            XOR            A                ;Set checksum = 0
2C30 4E             08720 WBLP       LD            C,(HL)            ;P/u byte
2C31 81             08730            ADD            A,C               ;Add checksum
2C32 F5             08740            PUSH           AF               ;Save A
2C33 CD512C        08750            CALL            WRBYTE           ;Write byte
2C36 F1             08760            POP            AF               ;Recover checksum
2C37 2C             08770            INC            L                ;Bump count
2C38 20F6           08780            JR            NZ,WBLP
                     08790 ;
                     08800 ;            Write checksum byte
                     08810 ;
2C3A ED44           08820            NEG                            ;Negate checksum
2C3C 4F             08830            LD            C,A               ;Write checksum byte
2C3D CD512C        08840            CALL            WRBYTE
                     08850 ;
                     08860 ;            Write 20 dummy bytes - X'00'
                     08870 ;
2C40 0614           08880            LD            B,20               ;Write 20 dummy zeroes
2C42 0E00           08890 WDLP       LD            C,0
2C44 CD512C        08900            CALL            WRBYTE
2C47 10F9           08910            DJNZ           WDLP
2C49 C9             08920            RET                            ;Done
                     08930 ;
                     08940 ;            WRBYTEC - Write a byte & add checksum
                     08950 ;
2C4A CD512C        08960 WRBYTEC CALL            WRBYTE            ;Write byte
2C4D 79             08970            LD            A,C               ;P/u byte
2C4E 82             08980            ADD            A,D               ;Add checksum
2C4F 57             08990            LD            D,A               ;New checksum
2C50 C9             09000            RET                            ;And RETURN
                     09010 ;
                     09020 ;            WRBYTE - Write a byte to Cassette
                     09030 ;            C => Byte to Output
                     09040 ;
2C51 C5             09050 WRBYTE: PUSH            BC                ;Save regs
2C52 D5             09060            PUSH            DE
2C53 CDDA2B        09070            CALL            NOPULS           ;Write dummy pulse
2C56 0608           09080            LD            B,8               ;8 bits to write
2C58 CDD12B        09090 WRBTLP    CALL            WRBIT            ;Write bit
2C5B 10FB           09100            DJNZ           WRBTLP
2C5D D1             09110            POP            DE               ;Restore regs
2C5E C1             09120            POP            BC
2C5F C9             09130            RET
                     09140 ;
                     09150 ;            WRSYNC - Write a SYNC pattern to Cassette
                     09160 ;
2C60 F3             09170 WRSYNC    DI                            ;Disable interrupts
2C61 C5             09180            PUSH            BC               ;Save BC
2C62 0680           09190            LD            B,80H            ;Delay

```

```

2C64      09200      @PAUSE
2C64 3E10      09045      LD      A,16
2C66 EF      09046      RST      40
2C67 015500      09210      LD      BC,0055H      ;B = 256, C = X'55'
                09220      ;
                09230      ;      Write SYNC bytes - X'55'
                09240      ;
2C6A CD762C      09250 WR55LP CALL      WRBYTE8      ;Write 8 bit byte
2C6D 10FB      09260      DJNZ     WR55LP
                09270      ;
                09280      ;      Write Marker byte - X'7F'
                09290      ;
2C6F 0E7F      09300      LD      C,7FH      ;Write marker byte X'7F'
2C71 CD762C      09310      CALL     WRBYTE8
2C74 C1      09320      POP      BC      ;Recover BC
2C75 C9      09330      RET      ;Done
                09340      ;
2C76 C5      09350 WRBYTE8 PUSH     BC      ;Save B
2C77 0608      09360      LD      B,8      ;8 bits long
2C79 CDD12B      09370 WB8LP  CALL     WRBIT      ;Write bit
2C7C 10FB      09380      DJNZ     WB8LP
2C7E C1      09390      POP      BC
2C7F C9      09400      RET
2C80      04290 *GET     TAPE100B:3
                09410 ;TAPE100B/ASM - Disk I/O & other routines
                09420 ;
                09430 ;      DISPSTR - Display String
                09440 ;
2C80 D5      09450 DISPSTR PUSH     DE      ;Save DE
2C81 ED5B6627      09460      LD      DE,(CURPOS) ;P/u cursor position
2C85 7E      09470 DSLP   LD      A,(HL)      ;P/u source char
2C86 FE03      09480      CP      ETX      ;Done ?
2C88 2815      09490      JR      Z,EXIT1      ;Yes - exit
2C8A FE0D      09500      CP      CR      ;Done ?
2C8C 280E      09510      JR      Z,EXIT2      ;Yes - exit
2C8E FE0A      09520      CP      LF      ;Line feed ?
2C90 2005      09530      JR      NZ,STUFCHR      ;No - stuff character
2C92 CDA52C      09540      CALL     NEXTLIN      ;Get next line
2C95 1802      09550      JR      BUMPIT
2C97 12      09560 STUFCHR LD      (DE),A      ;Output to video
2C98 13      09570      INC     DE
2C99 23      09580 BUMPIT INC     HL      ;No - bump count
2C9A 18E9      09590      JR      DSLP
2C9C CDA52C      09600 EXIT2  CALL     NEXTLIN      ;Next one down
2C9F ED536627      09610 EXIT1  LD      (CURPOS),DE ;Save cursor position
2CA3 D1      09620      POP      DE      ;Restore DE
2CA4 C9      09630      RET
                09640 ;
                09650 ;      NEXTLIN - Position to next line on video
                09660 ;      DE => RAM location
                09670 ;
2CA5 E5      09680 NEXTLIN PUSH     HL      ;Save regs
2CA6 EB      09690      EX      DE,HL      ;Xfer # to HL
2CA7 CDE32D      09700      CALL     GETCRS      ;Calculate X,Y
2CAA 24      09710      INC     H      ;Bump row #
2CAB 2E00      09720      LD      L,0      ; and start @ beginning
2CAD CDCF2D      09730      CALL     GETPOS2      ;Convert to RAM location
2CB0 EB      09740      EX      DE,HL      ;Stuff into DE
2CB1 E1      09750      POP      HL
2CB2 C9      09760      RET
                09770 ;

```

```

09780 ;          CONV_UC - Convert A to upper case
09790 ;
2CB3 FE61 09800 CONV_UC CP      'a'          ;Lower case ?
2CB5 D8    09810          RET      C          ;No
2CB6 FE7B 09820          CP      'z'+1        ;Lower case ?
2CB8 D0    09830          RET      NC         ;No
2CB9 CBAF 09840          RES      5,A         ;Convert to Upper Case
2CBB C9    09850          RET
09860 ;
09870 ;          CURSOFF - Turn off Cursor
09880 ;
2CBC F5    09890 CURSOFF PUSH    AF          ;Save regs
2CBD D5    09900          PUSH    DE
2CBE C5    09910          PUSH    BC
2CBF 0E0F 09920          LD      C,CUROFF     ;Cursor off Character
2CC1       09930          @@DSP
2CC1 3E02 09940          LD      A,2
2CC3 EF    09940          RST      40
2CC4 C24627 09940          JP      NZ,IOERR
2CC7 C1    09950          POP     BC          ;Restore regs
2CC8 D1    09960          POP     DE
2CC9 F1    09970          POP     AF
2CCA C9    09980          RET
09990 ;
10000 ;          INIT - Init a file
10010 ;
2CCB 3E3A 10020 INIT      LD      A,@INIT     ;SVC #
2CCD 1806 10030          JR      DOSVC       ;INIT file
10040 ;
10050 ;          OPEN - Open Source File
10060 ;
2CCF FDCB12C6 10070 OPEN  SET     0,(IY+SFLAG$) ;Inhibit file-open bit
2CD3 3E3B 10080          LD      A,@OPEN     ;OPEN SVC #
10090 ;
2CD5 F5    10100 DOSVC   PUSH    AF
2CD6 D5    10110          PUSH    DE
2CD7 219D27 10120          LD      HL,DFBUF   ;HL => Disk filename buf
2CDA 1A    10130 TLP     LD      A,(DE)      ;P/u byte from FCB
2CDB 77    10140          LD      (HL),A     ;Xfer to TEMBUF
2CDC 23    10150          INC     HL
2CDD 13    10160          INC     DE
2CDE FE0E 10170          CP      CR+1        ;Done ?
2CE0 3808 10180          JR      C,DUN
2CE2 FE3A 10190          CP      ':'
2CE4 2804 10200          JR      Z,DUN
2CE6 FE2E 10210          CP      '.'
2CE8 20F0 10220          JR      NZ,TLP
10230 ;
10240 ;          Found valid terminator - Is this a device ?
10250 ;
2CEA 2B    10260 DUN     DEC     HL          ;Back up to term
2CEB D1    10270          POP     DE          ;DE => FCB+0
2CEC 1A    10280          LD      A,(DE)     ;Device ?
2CED FE2A 10290          CP      '*'
2CEF 2807 10300          JR      Z,DUN2      ;Yes - done
2CF1 363A 10310          LD      (HL),':'    ;No - overwrite with ":"
2CF3 23    10320          INC     HL          ;Bump
2CF4 22122D 10330          LD      (DSPEC+1),HL ;Save drivespec location
2CF7 23    10340          INC     HL          ;Bump
2CF8 3603 10350 DUN2    LD      (HL),ETX    ;End with X'03'
2CFA F1    10360          POP     AF          ;A = SVC #

```

The Source	UTILITY Files	TAPE100 - LS-DOS 6.2	Page 00020
2CFB 32212D	10370	LD (SVCNUM+1),A	;Save SVC #
2CFE 21002F	10380	LD HL,I0BUFF	;HL => I/O Buffer
2D01 0600	10390	LD B,0	;LRL = 256
2D03 EF	10400	RST 28H	;OPEN or INIT file
2D04 2803	10410	CHECK JR Z,CHKPROT	;Check PROTection status
	10420 ;		
	10430 ;	Ignore Error #42 - "LRL Open Fault"	
	10440 ;		
2D06 FE2A	10450	CP 42	;Ignore this error
2D08 C0	10460	RET NZ	;NZ - Abort
	10470 ;		
	10480 ;	Stuff Drive # into Buffer	
	10490 ;		
2D09 D5	10500	CHKPROT PUSH DE	;P/u drivespec
2D0A DDE1	10510	POP IX	; from FCB+6
2D0C DD7E06	10520	LD A,(IX+6)	
2D0F C630	10530	ADD A,'0'	;Convert to ASCII
2D11 320000	10540	DSPEC LD (\$-\$),A	
	10550 ;		
	10560 ;	Check if File has proper Access	
	10570 ;		
2D14 DDCB007E	10580	BIT 7,(IX)	;Is FCB open?
2D18 281F	10590	JR Z,ILLFILE	;No - Illegal Filename
2D1A DD7E01	10600	LD A,(IX+1)	;P/u protection byte
2D1D E607	10610	AND 7	
2D1F 47	10620	LD B,A	;Xfer to B
	10630 ;		
2D20 3E00	10640	SVCNUM LD A,\$-\$;P/u SVC #
2D22 FE3A	10650	CP @INIT	;@INIT ?
2D24 78	10660	LD A,B	;P/u protection level
2D25 280C	10670	JR Z,INIT1	;Z - Must be < 5
2D27 FE06	10680	CP 6	;Read Access ?
2D29 380C	10690	JR C,OKYDOKY	;Yes - set Z & RETurn
	10700 ;		
	10710 ;	Illegal Access to protected file	
	10720 ;		
2D2B	10730	ILLACC @@CLOSE	;Close File
2D2B 3E3C	00049	LD A,60	
2D2D EF	00050	RST 40	
2D2E 3E19	10740	LD A,25	;File Access Denied
2D30 C34627	10750	JP IOERR	;Error - Regardless
	10760 ;		
2D33 FE05	10770	INIT1 CP 5	;Update Access ?
2D35 30F4	10780	JR NC,ILLACC	;No - Illegal Access
2D37 AF	10790	OKYDOKY XOR A	;RETurn Z
2D38 C9	10800	RET	
	10810 ;		
2D39 3E13	10820	ILLFILE LD A,19	;Illegal Filename
2D3B B7	10830	OR A	;Set NZ
2D3C C9	10840	RET	
	10850 ;		
	10860 ;	CLOSE - Close the Destination File	
	10870 ;		
2D3D 11112E	10880	CLOSE LD DE,FCB2	;DE => FCB
2D40	10890	@@CLOSE	;Close File
2D40 3E3C	00051	LD A,60	
2D42 EF	00052	RST 40	
2D43 C8	10900	RET Z	;Good - RETurn
2D44 C34627	10910	JP IOERR	;Bad - Quit
	10920 ;		
	10930 ;	WRITESC - Write a Sector to Destination file	

```

10940 ;
2D47 11112E 10950 WRITESC LD DE,FCB2 ;DE => FCB
2D4A 10960 @WRITE ;Write Sector
2D4A 3E4B 00053 LD A,75
2D4C EF 00054 RST 40
2D4D C24627 10970 JP NZ,IOERR ;Bad - quit
2D50 C9 10980 RET ;Good - RETurn
10990 ;
11000 ; WRTDEST - Write Destination File
11010 ;
2D51 11112E 11020 WRTDEST LD DE,FCB2 ;DE => Destination FCB
2D54 21152E 11030 WRTDES LD HL,FCB2+4 ;HL => msb of I/O buffer
2D57 34 11040 INC (HL) ;Bump
2D58 CD472D 11050 CALL WRITESC ;Write Sector
2D5B 3E00 11060 EOTF2 LD A,$-$ ;P/u # of sectors
2D5D BE 11070 CP (HL) ;Finished ?
2D5E 20F4 11080 JR NZ,WRTDES ;No - back to loop
11090 ;
11100 ; Finished Writing - Set EOF offset byte
11110 ;
2D60 3E00 11120 OFFSET LD A,$-$ ;P/u offset byte
2D62 32192E 11130 LD (FCB2+8),A ; & stuff into FCB
2D65 CD3D2D 11140 CALL CLOSE ;Close the File
2D68 C9 11150 RET
11160 ;
11170 ; READSRC - Read in chunk of Source Disk file
11180 ;
2D69 21F52D 11190 READSRC LD HL,FCB1+4 ;HL => Hi byte of I/O buf
2D6C 362F 11200 LD (HL),MEM<-8-1 ;Init FCB I/O buffer
11210 ;
11220 ; Read in Source file
11230 ;
2D6E 11F12D 11240 READSR2 LD DE,FCB1 ;Pt DE to FCB
2D71 34 11250 INC (HL) ;Bump I/O buffer
2D72 11260 @READ ;Read a sector
2D72 3E43 00055 LD A,67
2D74 EF 00056 RST 40
2D75 28F7 11270 JR Z,READSR2
11280 ;
11290 ; Fill remainder of sector w/ X'1A's
11300 ;
2D77 F5 11310 PUSH AF ;Save Error code
2D78 3AF92D 11320 NOMORE LD A,(FCB1+8) ;P/u EOF offset byte
2D7B ED44 11330 NEG
2D7D 47 11340 LD B,A ;Xfer to B for DJNZ
2D7E 66 11350 LD H,(HL) ;P/u I/O buffer msb
2D7F 2EFF 11360 LD L,0FFH ;End of sector
2D81 2801 11370 JR Z,NULBUF ;Z - keep HL here
2D83 25 11380 DEC H ;Sector boundary
2D84 361A 11390 NULBUF LD (HL),1AH ;Fill remainder of buffer
2D86 2B 11400 DEC HL ; with zeroes
2D87 10FB 11410 DJNZ NULBUF
11420 ;
11430 ; Add a sector of 1As
11440 ;
2D89 24 11450 INC H ;Pt to next sector
2D8A 2E00 11460 LD L,0
2D8C 361A 11470 XTR1AS LD (HL),01AH ;EOF indicator
2D8E 23 11480 INC HL ;Bump
2D8F 10FB 11490 DJNZ XTR1AS
2D91 F1 11500 DONTFIL POP AF ;Recover error code

```

```

11510 ;
11520 ; I/O Error - Better be EOF error
11530 ;
2D92 FE1C 11540 CP 1CH ;EOF ?
2D94 C8 11550 RET Z ;Yes - RETURN
2D95 FE1D 11560 CP 1DH ;NRN > ERN
2D97 C8 11570 RET Z ;Yes - RETURN
2D98 C34627 11580 JP IOERR ;No - Disk Error
11590 ;
11600 ; ENDOKI - Enable Video & Keyboard
11610 ;
2D9B F5 11620 ENDOKI PUSH AF
2D9C E5 11630 PUSH HL
2D9D 3A7800 11640 LD A,(OPREG$) ;P/u port mask
2DA0 32AC2D 11650 LD (SVOPREG+1),A ; and save it for DISDOKI
2DA3 CB87 11660 RES 0,A ;Reset bit 0
2DA5 CBCF 11670 SET 1,A ;Set bit 1
2DA7 1804 11680 JR DOOPREG ;Set new assignment
11690 ;
11700 ; DISDOKI - Disable Video & Keyboard
11710 ;
2DA9 F5 11720 DISDOKI PUSH AF
2DAA E5 11730 PUSH HL
11740 ;
2DAB 3E00 11750 SVOPREG LD A,$-$ ;Restore original mask
2DAD 327800 11760 DOOPREG LD (OPREG$),A
2DB0 D384 11770 OUT (@OPREG),A ; and disable video
11780 ;
2DB2 E1 11790 POP HL ;Restore regs & RETURN
2DB3 F1 11800 POP AF
2DB4 C9 11810 RET
11820 ;
11830 ; SWAP38 - Swap 38H - 3AH with save area
11840 ;
2DB5 0603 11850 SWAP38 LD B,3 ;3 bytes to exchange
2DB7 21C72D 11860 LD HL,SWAREA ;HL => Swap Area
2DBA 113800 11870 LD DE,38H ;DE => Restart Xfer addr
2DBD 4E 11880 SWAPLP LD C,(HL) ;P/u source
2DBE 1A 11890 LD A,(DE)
2DBF EB 11900 EX DE,HL ;Swap ptrs
2DC0 71 11910 LD (HL),C ;Stuff in dest
2DC1 12 11920 LD (DE),A
2DC2 23 11930 INC HL ;Bump ptrs
2DC3 13 11940 INC DE
2DC4 10F7 11950 DJNZ SWAPLP ;3 bytes to swap
2DC6 C9 11960 RET
11970 ;
2DC7 C3B02A 11980 SWAREA JP RST38V ;JP vector
11990 ;
12000 ; GETPOS - Get current cursor position in video
12010 ;
2DCA 0604 12020 GETPOS LD B,4 ;P/u current cursor pos
2DCC 12030 @@VDCTL
2DCC 3E0F 00057 LD A,15
2DCE EF 00058 RST 40
2DCF 4D 12040 GETPOS2 LD C,L ;Save column #
2DD0 6C 12050 LD L,H
2DD1 2600 12060 LD H,0 ;HL => Row #
2DD3 54 12070 LD D,H ;Set DE = HL
2DD4 5D 12080 LD E,L
2DD5 29 12090 ADD HL,HL ;X 2

```

```

2DD6 29      12100      ADD      HL,HL          ;X 4
2DD7 19      12110      ADD      HL,DE          ;X 5
2DD8 29      12120      ADD      HL,HL          ;X 10
2DD9 29      12130      ADD      HL,HL          ;X 20
2DDA 29      12140      ADD      HL,HL          ;X 40
2ddb 29      12150      ADD      HL,HL          ;X 80
2DDC 06F8    12160      LD        B,VIDEO<-8    ;D = high byte of video
2DDE 09      12170      ADD      HL,BC          ;HL => Cursor location
2DDF 226627  12180      LD        (CURPOS),HL    ;Save cursor position
2DE2 C9      12190      RET

                12200 ;
                12210 ;      GETCRS - Calculate row x column cursor pos
                12220 ;      HL => Cursor position in RAM
                12230 ;      HL <= Cursor position in Row (H) Column (L)
                12240 ;

2DE3 1100F8  12250      GETCRS LD        DE,VIDEO    ;Get offset
2DE6 B7      12260      OR        A
2DE7 ED52    12270      SBC      HL,DE
2DE9 0E50    12280      LD        C,80              ;Calculate row #
2DEB        12290      @@DIV16
2DEB 3E5E    00059      LD        A,94
2DED EF      00060      RST      40
2DEE 65      12300      LD        H,L              ;Set H = Row
2DEF 6F      12310      LD        L,A              ;Set L = Column
2DF0 C9      12320      RET

                04300 ;
0020        04310      FCB1    DS        32
0020        04320      FCB2    DS        32
0019        04330      INBUFF  DS        25
                04340 ;

2F00        04350      ORG     $<-8+1<+8
                04360 ;
0100        04370      IOBUFF  DS        256
3000        04380      MEM     EQU     $
                04390 ;
2600        04400      END     START

```

@@1	0000 @@2	0000 @@3	0000
@@4	0000 @INIT	003A @MOD2	0000
@MOD4	FFFF @OPEN	003B @OPREG	0084
ABB	0010 ABORT	2754 ABSVAL	2B65
AFTER	2A73 AP	0027 BIT0LOW	2AC1
BIT1	2B6C BIT1LOW	2AC5 BOGUS	2C04
BOGUSLP	2AF6 BREAK	0080 BREAKLC	F440
BS	0008 BT1	2BD5 BUFFER	2797
BUMPIT	2C99 CASSOFF	2A2B CASSON	2A1A
CFLAG\$	0002 CHECK	2D04 CHKERR	2B23
CHKERR2	2B27 CHKMAR	2B7E CHKPRM	261A
CHKPROT	2D09 CHKSEC	26C6 CIOERR	2B9C
CKFILE	2B16 CKPLP	27E9 CLEAN	27B2
CLOSE	2D3D CONV UC	2CB3 CORRECT	2B0D
COUNT	2852 CPARM	27FD CR	000D
CRESP	29E0 CUROFF	000F CURON	000E
CURPOS	2766 CURSOFF	2CBC DDF	2941
DELAY0	2B2F DELAY1	1217 DEL LP	2BDD
DEL LP2	2BE4 DFBUF	279D DFLAG\$	0003
DIFFER	000D DISDOKI	2DA9 DISPSTR	2C80
DLEN	2763 DODJ	2B6D DOINIT	27C7
DOINPUT	2816 DONTFIL	2D91 DOOPREG	2DAD
DOSVC	2CD5 DSF	290A DSLP	2C85
DSP	2843 DSPEC	2D11 DSPLY	2849
DUMBYT	2B08 DUMMY	2C13 DUN	2CEA
DUN2	2CF8 DUNLIN	27FC ENDOKI	2D9B
ENUF	2660 EOTF	2A51 EOTF2	2D5B
ETX	0003 EXDSP	284D EXIT	2758
EXIT1	2C9F EXIT2	2C9C FCB1	2DF1
FCB2	2E11 FILELP	2BFB FILENM	2790
FLAG	0040 FORNOW	273A GETBIT	2B80
GETCRS	2DE3 GETFILN	27BA GETPOS	2DCA
GETPOS2	2DCF GTFIL	27AC GTFIL2	26E5
HELLO\$	2853 ILLACC	2D2B ILLEGAL	2751
ILLFILE	2D39 INBUFF	2E31 INIT	2CCB
INIT1	2D33 INPUT	2835 INP R W	2621
IOBUFF	2F00 IOERR	2746 KFLAG\$	000A
LF	000A LOADA	003A LOOP	2B55
MEM	3000 MODMASK	000C MODOUT	00EC
NEXTINS	2BCE NEXTLIN	2CA5 NOMORE	2D78
NOPULS	2BDA NOTBLNK	2BCB NOTENT	2A41
NULBUF	2D84 NUM	0080 OFFSET	2D60
OKYDOKY	2D37 OLDSP	275B OPDSRC	264F
OPEN	2CCF OPREG\$	0078 PARMTBL	29C8
PAR ERR	002C PORTE0	00E0 PORTFF	00FF
PRDEST	2810 PRDEST2	282D PRMERR	29BB
PRMERR\$	297F PRSOUR	2808 PRSOUR2	2825
PRTAPE	2A3B RBLP	2A99 RBTLP	2B42
RDBIT	2A96 RDBLP	2BB0 RDBYTE	2BA7
RDBYTEC	2BA2 RDDAT	2A49 RDDAT2	2A4C
RDDATA	2A58 RDHEAD	2AD8 RDLP1	2A66
RDLP2	2A8D RDORWR	28DB RDSYNC	2B39
RDSYNC2	2B40 RDTAPE	26B1 READERR	2768
READFIL	26EB READING	277B READSR2	2D6E
READSRC	2D69 RESCNT	2B50 RFNLP	2AEB
ROTBYTE	2B8F ROUTOFF	0006 RPARM	29E4
RRESP	29D7 RST38V	2AB0 SFLAG\$	0012
SKPSPC	26D4 START	2600 STARTA	2609
STR	0020 STUFCHR	2C97 SVCNUM	2D20
SVOPREG	2DAB SWAP38	2DB5 SWAPLP	2BDD

SWAREA	2DC7	TAB	0009	TDF	2923
TLP	2CDA	TOOBIG	29BF	TOOBIG\$	2990
TOOLONG	003E	TOOSHRT	000F	TREADY	295F
TSF	28F1	USEHI	27D6	VFLAG\$	0015
VIDEO	F800	WAITINT	2AD5	WB8LP	2C79
WBLP	2C30	WDLP	2C42	WHICH1	0022
WPARM	29E6	WR55LP	2C6A	WRBIT	2BD1
WRBTLP	2C58	WRBYTE	2C51	WRBYTE8	2C76
WRBYTEC	2C4A	WRDAT	2C19	WRDAT2	2C1C
WRDATA	2C27	WRESP	29CF	WRHEAD	2BEC
WRITESC	2D47	WRITING	2785	WRMASK	0016
WRSYNC	2C60	WRTAPE	263A	WRTAPE2	2643
WRTDES	2D54	WRTDES2	2740	WRTDEST	2D51
XTRLAS	2D8C	@@ABORT	926F	@@ADTSK	9302
@@BANK	981A	@@BKSP	94FA	@@BREAK	9830
@@CHNIO	925A	@@CKBRKC	987E	@@CKDRV	9356
@@CKEOF	950F	@@CKTSK	92ED	@@CLOSE	94E5
@@CLS	9868	@@CMNDI	9299	@@CMNDR	92AE
@@CTL	90BE	@@DATE	9230	@@DCSTAT	9395
@@DEBUG	92D8	@@DECHEX	979A	@@DIRRD	9707
@@DIRWR	971C	@@DIV16	9785	@@DIV8	9770
@@DODIR	936B	@@DSP	9082	@@DSPLY	9122
@@ERROR	92C3	@@EXIT	9284	@@FEXT	9674
@@FLAGS	9804	@@FNAME	9689	@@FSPEC	965F
@@GATRD	96F2	@@GATWR	9731	@@GET	9096
@@GTDCB	96B3	@@GTDCT	969E	@@GTMOD	96C8
@@HDFMT	943D	@@HEX16	97D9	@@HEX8	97C4
@@HEXDEC	97AF	@@HIGH\$	97EE	@@INIT	94BB
@@KBD	90FA	@@KEY	906E	@@KEYIN	910E
@@KLTSK	9341	@@LOAD	9635	@@LOC	9524
@@LOF	9539	@@LOGGER	9159	@@LOGOT	916E
@@MSG	91A5	@@MUL16	975B	@@MUL8	9746
@@OPEN	94D0	@@PARAM	921B	@@PAUSE	9206
@@PEOF	954E	@@POSN	9563	@@PRINT	91BA
@@PRT	90D2	@@PUT	90AA	@@RAMDIR	9380
@@RDSEC	9413	@@RDSSC	96DD	@@READ	9578
@@REMOV	94A6	@@RENAM	9491	@@REW	958D
@@RMTSK	9317	@@RPTSK	932C	@@RREAD	95A2
@@RSLCT	93FE	@@RSTOR	93BF	@@RUN	964A
@@RWRTIT	95B7	@@SEEK	93E9	@@SEEKSC	95CC
@@SKIP	95E1	@@SLCT	93AA	@@STEPI	93D4
@@TIME	9245	@@VDCTL	91F1	@@VER	95F6
@@VRSEC	9428	@@WEOF	960B	@@WHERE	90E6
@@WRITE	9620	@@WRSEC	9452	@@WRSSC	9467
@@WRTRK	947C				

2600 is the transfer address

00000 Total errors

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:

NOTES:


```

00100 ;LDOS60/EQU - Equates from cross reference of Lowcore
00110 TITLE <LDOS60/EQU>
00120 ;
08F0 00130 @$SYS EQU 08F0H
0000 00140 @01 DEFL 0000H
0000 00150 @02 DEFL 0000H
0000 00160 @03 DEFL 0000H
0000 00170 @04 DEFL 0000H
0877 00180 @BANK EQU 0877H
1300 00190 @BYTEIO EQU 1300H
0689 00200 @CHNIO EQU 0689H
0553 00210 @CKBRKC EQU 0553H
0545 00220 @CLS EQU 0545H
0623 00230 @CTL EQU 0623H
07A8 00240 @DATE EQU 07A8H
06E3 00250 @DIV16 EQU 06E3H
0642 00260 @DSP EQU 0642H
052D 00270 @DSPLY EQU 052DH
0000 00280 @FRENCH EQU 0000H
0000 00290 @GERMAN EQU 0000H
0638 00300 @GET EQU 0638H
07BD 00310 @HEX16 EQU 07BDH
07C2 00320 @HEX8 EQU 07C2H
06F6 00330 @HEXDEC EQU 06F6H
0000 00340 @HZ50 EQU 0000H
0000 00350 @INTL EQU 0000H
0630 00360 @JCL EQU 0630H
0635 00370 @KBD EQU 0635H
0628 00380 @KEY EQU 0628H
0585 00390 @KEYIN EQU 0585H
0089 00400 @KITSK EQU 0089H
0503 00410 @LOGGER EQU 0503H
0500 00420 @LOGOT EQU 0500H
0000 00430 @MOD2 EQU 0000H
FFFF 00440 @MOD4 EQU 0FFFFH
0530 00450 @MSG EQU 0530H
06C9 00460 @MUL16 EQU 06C9H
0084 00470 @OPREG EQU 0084H
0528 00480 @PRINT EQU 0528H
063D 00490 @PRT EQU 063DH
0645 00500 @PUT EQU 0645H
0FE9 00510 @RSTNMI EQU 0FE9H
0680 00520 @RSTREG EQU 0680H
078D 00530 @TIME EQU 078DH
FFFF 00540 @USA EQU 0FFFFH
0B99 00550 @VDCTL EQU 0B99H
0D38 00560 @VDCTL3 EQU 0D38H
0D42 00570 @ VDCTL EQU 0D42H
0DF1 00580 ADDR_2 ROWCOL EQU 0DF1H
0201 00590 BAR$ EQU 0201H
439D 00600 BOOTST$ EQU 439DH
0200 00610 BUR$ EQU 0200H
0A7B 00620 CASHK$ EQU 0A7BH
006C 00630 CFLAG$ EQU 006CH
0300 00640 CORE$ DEFL 0300H
F800 00650 CRTBGN$ EQU 0F800H
0033 00660 DATE$ EQU 0033H
04C7 00670 DAYTBL$ EQU 04C7H
0031 00680 DCBKL$ EQU 0031H
0470 00690 DCT$ EQU 0470H
006D 00700 DFLAG$ EQU 006DH

```

The Source	UTILITY Files	LDOS60/EQU	Page 00002
0846	00710 DIS DO RAM	EQU 0846H	
0B94	00720 DODATA\$ EQU	0B94H	
0210	00730 DODCB\$ EQU	0210H	
0C44	00740 DO_CONTROL	EQU 0C44H	
0CB8	00750 DO_DSPCHAR	EQU 0CB8H	
0C8C	00760 DO_INVERT_DIS	EQU 0C8CH	
0C89	00770 DO_INVERT_ENA	EQU 0C89H	
0C9B	00780 DO_INVERT_OFF	EQU 0C9BH	
0000	00790 DO_MASK EQU	0000H	
0BCB	00800 DO_RET EQU	0BCBH	
0BCC	00810 DO_RET1 EQU	0BCCH	
0CCE	00820 DO_SCROLL	EQU 0CCEH	
0BEA	00830 DO_TABS EQU	0BEAH	
04C0	00840 DSKTYP\$ EQU	04C0H	
04C2	00850 DTPMT\$ EQU	04C2H	
0FF4	00860 DVREND\$ EQU	0FF4H	
0206	00870 DVRHI\$ EQU	0206H	
0817	00880 ENADIS DO RAM	EQU 0817H	
000E	00890 FDDINT\$ EQU	000EH	
006A	00900 FLGTAB\$ EQU	006AH	
0DAE	00910 GET @ ROWCOL	EQU 0DAEH	
0750	00920 HERTZ\$ EQU	0750H	
040E	00930 HIGH\$ EQU	040EH	
0072	00940 IFLAG\$ EQU	0072H	
0420	00950 INBUF\$ EQU	0420H	
003E	00960 INTVC\$ EQU	003EH	
0203	00970 JCLCB\$ EQU	0203H	
0230	00980 JLDCB\$ EQU	0230H	
07D6	00990 KCK@ EQU	07D6H	
0074	01000 KFLAG\$ EQU	0074H	
08FC	01010 KIDATA\$ EQU	08FCH	
0208	01020 KIDCB\$ EQU	0208H	
0202	01030 LBANK\$ EQU	0202H	
0401	01040 MAXDAY\$ EQU	0401H	
0076	01050 MODOUT\$ EQU	0076H	
04DC	01060 MONTBL\$ EQU	04DCH	
0077	01070 NFLAG\$ EQU	0077H	
0078	01080 OPREG\$ EQU	0078H	
086E	01090 OPREG_SV AREA	EQU 086EH	
0835	01100 OPREG_SV_PTR	EQU 0835H	
0410	01110 PAKNAM\$ EQU	0410H	
0382	01120 PAUSE@ EQU	0382H	
07AF	01130 PCSAVE\$ EQU	07AFH	
001B	01140 PDRV\$ EQU	001BH	
0218	01150 PRDCB\$ EQU	0218H	
0DCD	01160 PUTA@DE EQU	0DCDH	
0DCA	01170 PUT @ EQU	0DCAH	
0DC6	01180 PUT @ ROWCOL	EQU 0DC6H	
007B	01190 RFLAG\$ EQU	007BH	
0DD0	01200 ROWCOL 2 ADDR	EQU 0DD0H	
04C4	01210 RSTOR\$ EQU	04C4H	
0238	01220 SIDCB\$ EQU	0238H	
0CF3	01230 SET_SCROLL	EQU 0CF3H	
007C	01240 SFLAG\$ EQU	007CH	
0220	01250 SIDCB\$ EQU	0220H	
0228	01260 SODCB\$ EQU	0228H	
0380	01270 STACK\$ EQU	0380H	
0000	01280 START\$ EQU	0000H	
002D	01290 TIME\$ EQU	002DH	
002C	01300 TIMER\$ EQU	002CH	
002B	01310 TIMSL\$ EQU	002BH	

0713	01320	TIMTSK\$ EQU	0713H
04C3	01330	TMPMT\$ EQU	04C3H
07B1	01340	TRACE INT	EQU 07B1H
0A8F	01350	TYPHK\$ EQU	0A8FH
0B26	01360	TYPTSK\$ EQU	0B26H
007F	01370	VFLAG\$ EQU	007FH
0401	01380	ZERO\$ EQU	0401H

No end statement

00000 Total errors


```

00100 ;SYS0/EQU - Equates from cross reference of Sysres
00000 00110 TITLE <SYS0/EQU>
00120 ;
03B7 00130 $A1 EQU 03B7H
03B8 00140 $A2 EQU 03B8H
03B9 00150 $A3 EQU 03B9H
1470 00160 $CKEOF EQU 1470H
08F0 00170 @$SYS EQU 08F0H
0000 00180 @01 DEFL 0000H
0000 00190 @01 DEFL 0000H
0000 00200 @02 DEFL 0000H
0000 00210 @02 DEFL 0000H
0000 00220 @03 DEFL 0000H
0000 00230 @03 DEFL 0000H
0000 00240 @04 DEFL 0000H
0000 00250 @04 DEFL 0000H
1B08 00260 @ABORT EQU 1B08H
1CDA 00270 @ADTSK EQU 1CDAH
0877 00280 @BANK EQU 0877H
1486 00290 @BKSP EQU 1486H
196F 00300 @BREAK EQU 196FH
1300 00310 @BYTEIO EQU 1300H
0689 00320 @CHNIO EQU 0689H
0553 00330 @CKBRKC EQU 0553H
1993 00340 @CKDRV EQU 1993H
158F 00350 @CKEOF EQU 158FH
1CF5 00360 @CKTSK EQU 1CF5H
1999 00370 @CLOSE EQU 1999H
0545 00380 @CLS EQU 0545H
197E 00390 @CMNDI EQU 197EH
197B 00400 @CMNDR EQU 197BH
0623 00410 @CTL EQU 0623H
07A8 00420 @DATE EQU 07A8H
199F 00430 @DBGHK EQU 199FH
19C0 00440 @DCINIT EQU 19C0H
19C4 00450 @DCRES EQU 19C4H
19B5 00460 @DCSTAT EQU 19B5H
1A2B 00470 @DCTBYT EQU 1A2BH
19A0 00480 @DEBUG EQU 19A0H
03E1 00490 @DECHEX EQU 03E1H
18F7 00500 @DIRCYL EQU 18F7H
18BB 00510 @DIRRD EQU 18BBH
1803 00520 @DIRWR EQU 1803H
06E3 00530 @DIV16 EQU 06E3H
1927 00540 @DIV8 EQU 1927H
19AF 00550 @DODIR EQU 19AFH
19A9 00560 @DOKEY EQU 19A9H
0642 00570 @DSP EQU 0642H
052D 00580 @DSPLY EQU 052DH
1B0F 00590 @ERROR EQU 1B0FH
1B0B 00600 @EXIT EQU 1B0BH
1984 00610 @FEXT EQU 1984H
196A 00620 @FLAGS EQU 196AH
199C 00630 @FNAME EQU 199CH
0000 00640 @FRENCH EQU 0000H
1981 00650 @FSPEC EQU 1981H
1874 00660 @GATRD EQU 1874H
1875 00670 @GATWR EQU 1875H
0000 00680 @GERMAN EQU 0000H
0638 00690 @GET EQU 0638H
1990 00700 @GTDCB EQU 1990H

```

1A1E	00710 @GTDCT	EQU	1A1EH
19B2	00720 @GTMOD	EQU	19B2H
19E4	00730 @HDFMT	EQU	19E4H
07BD	00740 @HEX16	EQU	07BDH
07C2	00750 @HEX8	EQU	07C2H
06F6	00760 @HEXDEC	EQU	06F6H
1948	00770 @HIGH\$	EQU	1948H
1897	00780 @HITRD	EQU	1897H
1898	00790 @HITWR	EQU	1898H
0000	00800 @HZ50	EQU	0000H
0086	00810 @ICNFG	EQU	0086H
198D	00820 @INIT	EQU	198DH
0000	00830 @INTL	EQU	0000H
1BF2	00840 @IPL	EQU	1BF2H
0630	00850 @JCL	EQU	0630H
0635	00860 @KBD	EQU	0635H
0628	00870 @KEY	EQU	0628H
0585	00880 @KEYIN	EQU	0585H
0089	00890 @KITSK	EQU	0089H
0089	00900 @KITSK	EQU	0089H
1CD0	00910 @KLTSK	EQU	1CD0H
1B38	00920 @LOAD	EQU	1B38H
14B3	00930 @LOC	EQU	14B3H
14DE	00940 @LOF	EQU	14DEH
0503	00950 @LOGGER	EQU	0503H
0500	00960 @LOGOT	EQU	0500H
0000	00970 @MOD2	EQU	0000H
FFFF	00980 @MOD4	EQU	FFFFFH
0530	00990 @MSG	EQU	0530H
06C9	01000 @MUL16	EQU	06C9H
190A	01010 @MUL8	EQU	190AH
0066	01020 @NMI	EQU	0066H
198A	01030 @OPEN	EQU	198AH
0084	01040 @OPREG	EQU	0084H
1987	01050 @PARAM	EQU	1987H
0382	01060 @PAUSE	EQU	0382H
14A2	01070 @PEOF	EQU	14A2H
1434	01080 @POSN	EQU	1434H
0528	01090 @PRINT	EQU	0528H
063D	01100 @PRT	EQU	063DH
0645	01110 @PUT	EQU	0645H
19AC	01120 @RAMDIR	EQU	19ACH
19D8	01130 @RDHDR	EQU	19D8H
19F4	01140 @RDSEC	EQU	19F4H
18D8	01150 @RDSSC	EQU	18D8H
19E0	01160 @RDTRK	EQU	19E0H
1513	01170 @READ	EQU	1513H
19A6	01180 @REMOVE	EQU	19A6H
1996	01190 @RENAME	EQU	1996H
149B	01200 @REW	EQU	149BH
1CD7	01210 @RMITSK	EQU	1CD7H
1CEB	01220 @RPTSK	EQU	1CEBH
1473	01230 @RREAD	EQU	1473H
19D4	01240 @RSLCT	EQU	19D4H
0000	01250 @RST00	EQU	0000H
0008	01260 @RST08	EQU	0008H
0010	01270 @RST10	EQU	0010H
0018	01280 @RST18	EQU	0018H
0020	01290 @RST20	EQU	0020H
0028	01300 @RST28	EQU	0028H
0030	01310 @RST30	EQU	0030H

0038	01320 @RST38 EQU	0038H
0FE9	01330 @RSTNMI EQU	0FE9H
19C8	01340 @RSTOR EQU	19C8H
0680	01350 @RSTREG EQU	0680H
1B1D	01360 @RUN EQU	1B1DH
13AD	01370 @RWRT EQU	13ADH
19D0	01380 @SEEK EQU	19D0H
1421	01390 @SEEKSC EQU	1421H
1430	01400 @SKIP EQU	1430H
19BC	01410 @SLCT EQU	19BCH
0392	01420 @SOUND EQU	0392H
19CC	01430 @STEPI EQU	19CCH
078D	01440 @TIME EQU	078DH
FFFF	01450 @USA EQU	0FFFFH
0B99	01460 @VDCTL EQU	0B99H
0D38	01470 @VDCTL3 EQU	0D38H
1560	01480 @VER EQU	1560H
19DC	01490 @VRSEC EQU	19DCH
14EC	01500 @WEOF EQU	14ECH
1979	01510 @WHERE EQU	1979H
1531	01520 @WRITE EQU	1531H
19E8	01530 @WRSEC EQU	19E8H
19EC	01540 @WRSSC EQU	19ECH
19F0	01550 @WRTRK EQU	19F0H
0D42	01560 @ VDCTL EQU	0D42H
0DF1	01570 ADDR 2 ROWCOL EQU	0DF1H
006A	01580 AFLAG\$ EQU	006AH
1FF1	01590 AUTO? EQU	1FF1H
0201	01600 BAR\$ EQU	0201H
439D	01610 BOOTST\$ EQU	439DH
1C60	01620 BREAK? EQU	1C60H
1C88	01630 BRKVEC\$ EQU	1C88H
0200	01640 BUR\$ EQU	0200H
0A7B	01650 CASHK\$ EQU	0A7BH
00E0	01660 CFCB\$ EQU	00E0H
00E0	01670 CFGFCB\$ EQU	00E0H
006C	01680 CFLAG\$ EQU	006CH
006C	01690 CFLAG\$ EQU	006CH
1A7F	01700 CKMOD@ EQU	1A7FH
1568	01710 CKOPEN@ EQU	1568H
203F	01720 CONF IG\$ EQU	203FH
1CFF	01730 CORE\$ DEFL	1CFFH
1BFF	01740 CORE\$ DEFL	1BFFH
1948	01750 CORE\$ DEFL	1948H
1948	01760 CORE\$ DEFL	1948H
0300	01770 CORE\$ DEFL	0300H
F800	01780 CRTBGN\$ EQU	0F800H
16AE	01790 CYL_GRN EQU	16AEH
1A26	01800 D0F8BYT8 EQU	1A26H
0033	01810 DATE\$ EQU	0033H
0033	01820 DATE\$ EQU	0033H
04C7	01830 DAYTBL\$ EQU	04C7H
00A0	01840 DBGSV\$ EQU	00A0H
0031	01850 DCBKLS EQU	0031H
0470	01860 DCT\$ EQU	0470H
1A29	01870 DCTBYT8@ EQU	1A29H
1A34	01880 DCTFLD@ EQU	1A34H
006D	01890 DFLAG\$ EQU	006DH
006D	01900 DFLAG\$ EQU	006DH
2300	01910 DIRBUF\$ EQU	2300H
0846	01920 DIS_DO_RAM EQU	0846H

0B94	01930	DODATA\$ EQU	0B94H
0210	01940	DODCB\$ EQU	0210H
0C44	01950	DO_CONTROL EQU	0C44H
0CB8	01960	DO_DSPCHAR EQU	0CB8H
0C8C	01970	DO_INVERT_DIS EQU	0C8CH
0C89	01980	DO_INVERT_ENA EQU	0C89H
0C9B	01990	DO_INVERT_OFF EQU	0C9BH
0000	02000	DO_MASK EQU	0000H
0BCB	02010	DO_RET EQU	0BCBH
0BCC	02020	DO_RET1 EQU	0BCCH
0CCE	02030	DO_SCROLL EQU	0CCEH
0BEA	02040	DO_TABS EQU	0BEAH
04C0	02050	DSKTYP\$ EQU	04C0H
04C2	02060	DTPMT\$ EQU	04C2H
0FF4	02070	DVREND\$ EQU	0FF4H
0206	02080	DVRHI\$ EQU	0206H
006E	02090	EFLAG\$ EQU	006EH
0817	02100	ENADIS DO_RAM EQU	0817H
19A4	02110	EXTDBG\$ EQU	19A4H
000E	02120	FDDINT\$ EQU	000EH
000E	02130	FDDINT\$ EQU	000EH
006F	02140	FEMSK\$ EQU	006FH
006A	02150	FLGTAB\$ EQU	006AH
006A	02160	FLGTAB\$ EQU	006AH
0DAE	02170	GET @ ROWCOL EQU	0DAEH
0750	02180	HERTZ\$ EQU	0750H
040E	02190	HIGH\$ EQU	040EH
1A6C	02200	HKRES\$ EQU	1A6CH
0072	02210	IFLAG\$ EQU	0072H
0072	02220	IFLAG\$ EQU	0072H
0420	02230	INBUF\$ EQU	0420H
003C	02240	INTIM\$ EQU	003CH
003D	02250	INTMSK\$ EQU	003DH
003E	02260	INTVC\$ EQU	003EH
003E	02270	INTVC\$ EQU	003EH
0203	02280	JCLCB\$ EQU	0203H
0024	02290	JDCB\$ EQU	0024H
00C0	02300	JFCB\$ EQU	00C0H
0230	02310	JLDCB\$ EQU	0230H
0026	02320	JRET\$ EQU	0026H
07D6	02330	KCK@ EQU	07D6H
0074	02340	KFLAG\$ EQU	0074H
0074	02350	KFLAG\$ EQU	0074H
08FC	02360	KIDATA\$ EQU	08FCH
0208	02370	KIDCB\$ EQU	0208H
0202	02380	LBANK\$ EQU	0202H
0023	02390	LDRV\$ EQU	0023H
0075	02400	LFLAG\$ EQU	0075H
1566	02410	LNKFCB@ EQU	1566H
001E	02420	LOW\$ EQU	001EH
000D	02430	LSVC\$ EQU	000DH
2400	02440	MAXCOR\$ EQU	2400H
0401	02450	MAXDAY\$ EQU	0401H
3000	02460	MINCOR\$ EQU	3000H
0076	02470	MODOUT\$ EQU	0076H
0076	02480	MODOUT\$ EQU	0076H
04DC	02490	MONTBL\$ EQU	04DCH
0077	02500	NFLAG\$ EQU	0077H
0078	02510	OPREG\$ EQU	0078H
0078	02520	OPREG\$ EQU	0078H
086E	02530	OPREG_SV_AREA EQU	086EH

0835	02540 OPREG SV PTR	EQU	0835H
14DC	02550 ORARET@ EQU	14DCH	
003B	02560 OSRLS\$ EQU	003BH	
0085	02570 OSVER\$ EQU	0085H	
0069	02580 OVRLY\$ EQU	0069H	
0410	02590 PAKNAM\$ EQU	0410H	
0382	02600 PAUSE@ EQU	0382H	
07AF	02610 PCSAVE\$ EQU	07AFH	
001B	02620 PDRV\$ EQU	001BH	
001B	02630 PDRV\$ EQU	001BH	
001C	02640 PHIGH\$ EQU	001CH	
0218	02650 PRDCB\$ EQU	0218H	
0DCD	02660 PUTA@DE EQU	0DCDH	
0DCA	02670 PUT @ EQU	0DCAH	
0DC6	02680 PUT @ ROWCOL	EQU	0DC6H
007B	02690 RFLAG\$ EQU	007BH	
007B	02700 RFLAG\$ EQU	007BH	
0DD0	02710 ROWCOL 2 ADDR	EQU	0DD0H
1BFF	02720 RST38@ EQU	1BFFH	
04C4	02730 RSTOR\$ EQU	04C4H	
13A2	02740 RWRIT@ EQU	13A2H	
0238	02750 SIDCB\$ EQU	0238H	
1D00	02760 SBUFF\$ EQU	1D00H	
1A79	02770 SET@EXEC	EQU	1A79H
0CF3	02780 SET_SCROLL	EQU	0CF3H
008C	02790 SFCB\$ EQU	008CH	
007C	02800 SFLAG\$ EQU	007CH	
007C	02810 SFLAG\$ EQU	007CH	
0220	02820 SIDCB\$ EQU	0220H	
0228	02830 SODCB\$ EQU	0228H	
2142	02840 SPACE4\$ EQU	2142H	
0380	02850 STACK\$ EQU	0380H	
0000	02860 START\$ EQU	0000H	
0000	02870 START\$ EQU	0000H	
000B	02880 SVCRET\$ EQU	000BH	
0100	02890 SVCTAB\$ EQU	0100H	
1B13	02900 SYSERR\$ EQU	1B13H	
004E	02910 TCB\$ EQU	004EH	
007D	02920 TFLAG\$ EQU	007DH	
002D	02930 TIME\$ EQU	002DH	
002D	02940 TIME\$ EQU	002DH	
002C	02950 TIMER\$ EQU	002CH	
002C	02960 TIMER\$ EQU	002CH	
002B	02970 TIMSL\$ EQU	002BH	
002B	02980 TIMSL\$ EQU	002BH	
0713	02990 TIMTSK\$ EQU	0713H	
04C3	03000 TMPMT\$ EQU	04C3H	
07B1	03010 TRACE INT	EQU	07B1H
0A8F	03020 TYPHK\$ EQU	0A8FH	
0B26	03030 TYPTSK\$ EQU	0B26H	
0013	03040 USTOR\$ EQU	0013H	
007F	03050 VFLAG\$ EQU	007FH	
007F	03060 VFLAG\$ EQU	007FH	
0080	03070 WRINT\$ EQU	0080H	
0401	03080 ZERO\$ EQU	0401H	
13A0	03090 ZEROA@ EQU	13A0H	

No end statement

00000 Total errors


```

00100 ;SVC MAC/ASM - LS-DOS Version VI
0000 00110 TITLE <SVC MAC - MACRO EQUIVALENTS>
00120 ;*LIST OFF
00130 ;
0000 00140 @MOD2 EQU 0
FFFF 00150 @MOD4 EQU -1
0000 00160 @@KEY MACRO
0000 00170 LD A,1
0000 00180 RST 40
0000 00190 ENDM
0000 00200 @@DSP MACRO
0000 00210 LD A,2
0000 00220 RST 40
0000 00230 ENDM
0000 00240 @@GET MACRO
0000 00250 LD A,3
0000 00260 RST 40
0000 00270 ENDM
0000 00280 @@PUT MACRO
0000 00290 LD A,4
0000 00300 RST 40
0000 00310 ENDM
0000 00320 @@CTL MACRO
0000 00330 LD A,5
0000 00340 RST 40
0000 00350 ENDM
0000 00360 @@PRT MACRO
0000 00370 LD A,6
0000 00380 RST 40
0000 00390 ENDM
0000 00400 @@WHERE MACRO
0000 00410 LD A,7
0000 00420 RST 40
0000 00430 ENDM
0000 00440 @@KBD MACRO
0000 00450 LD A,8
0000 00460 RST 40
0000 00470 ENDM
0000 00480 @@KEYIN MACRO
0000 00490 LD A,9
0000 00500 RST 40
0000 00510 ENDM
0000 00520 @@DSPLY MACRO #MSG
0000 00530 IFEQ %,1
0000 00540 LD HL,#MSG
0000 00550 ENDIF
0000 00560 LD A,10
0000 00570 RST 40
0000 00580 ENDM
0000 00590 @@LOGGER MACRO
0000 00600 LD A,11
0000 00610 RST 40
0000 00620 ENDM
0000 00630 @@LOGOT MACRO #MSG
0000 00640 IFEQ %,1
0000 00650 LD HL,#MSG
0000 00660 ENDIF
0000 00670 LD A,12
0000 00680 RST 40
0000 00690 ENDM
0000 00700 @@MSG MACRO

```

0000	00710	LD	A,13
0000	00720	RST	40
0000	00730	ENDM	
0000	00740 @@PRINT	MACRO	#MSG
0000	00750	IFEQ	%,1
0000	00760	LD	HL,#MSG
0000	00770	ENDIF	
0000	00780	LD	A,14
0000	00790	RST	40
0000	00800	ENDM	
0000	00810 @@VDCTL	MACRO	
0000	00820	LD	A,15
0000	00830	RST	40
0000	00840	ENDM	
0000	00850 @@PAUSE	MACRO	
0000	00860	LD	A,16
0000	00870	RST	40
0000	00880	ENDM	
0000	00890 @@PARAM	MACRO	
0000	00900	LD	A,17
0000	00910	RST	40
0000	00920	ENDM	
0000	00930 @@DATE	MACRO	
0000	00940	LD	A,18
0000	00950	RST	40
0000	00960	ENDM	
0000	00970 @@TIME	MACRO	
0000	00980	LD	A,19
0000	00990	RST	40
0000	01000	ENDM	
0000	01010 @@CHNIO	MACRO	
0000	01020	LD	A,20
0000	01030	RST	40
0000	01040	ENDM	
0000	01050 @@ABORT	MACRO	
0000	01060	LD	A,21
0000	01070	RST	40
0000	01080	ENDM	
0000	01090 @@EXIT	MACRO	
0000	01100	LD	A,22
0000	01110	RST	40
0000	01120	ENDM	
0000	01130 @@CMNDI	MACRO	
0000	01140	LD	A,24
0000	01150	RST	40
0000	01160	ENDM	
0000	01170 @@CMNDR	MACRO	
0000	01180	LD	A,25
0000	01190	RST	40
0000	01200	ENDM	
0000	01210 @@ERROR	MACRO	
0000	01220	LD	A,26
0000	01230	RST	40
0000	01240	ENDM	
0000	01250 @@DEBUG	MACRO	
0000	01260	LD	A,27
0000	01270	RST	40
0000	01280	ENDM	
0000	01290 @@CKTSK	MACRO	
0000	01300	LD	A,28
0000	01310	RST	40

```

0000      01320      ENDM
0000      01330 @@ADTSK MACRO
0000      01340      LD      A,29
0000      01350      RST      40
0000      01360      ENDM
0000      01370 @@RMTSK MACRO
0000      01380      LD      A,30
0000      01390      RST      40
0000      01400      ENDM
0000      01410 @@RPTSK MACRO
0000      01420      LD      A,31
0000      01430      RST      40
0000      01440      ENDM
0000      01450 @@KLTSK MACRO
0000      01460      LD      A,32
0000      01470      RST      40
0000      01480      ENDM
0000      01490 @@CKDRV MACRO
0000      01500      LD      A,33
0000      01510      RST      40
0000      01520      ENDM
0000      01530 @@DODIR MACRO
0000      01540      LD      A,34
0000      01550      RST      40
0000      01560      ENDM
0000      01570 @@RAMDIR MACRO
0000      01580      LD      A,35
0000      01590      RST      40
0000      01600      ENDM
0000      01610 @@DCSTAT MACRO
0000      01620      LD      A,40
0000      01630      RST      40
0000      01640      ENDM
0000      01650 @@SLCT MACRO
0000      01660      LD      A,41
0000      01670      RST      40
0000      01680      ENDM
0000      01690 @@RSTOR MACRO
0000      01700      LD      A,44
0000      01710      RST      40
0000      01720      ENDM
0000      01730 @@STEPI MACRO
0000      01740      LD      A,45
0000      01750      RST      40
0000      01760      ENDM
0000      01770 @@SEEK MACRO
0000      01780      LD      A,46
0000      01790      RST      40
0000      01800      ENDM
0000      01810 @@RSLCT MACRO
0000      01820      LD      A,47
0000      01830      RST      40
0000      01840      ENDM
0000      01850 @@RDSEC MACRO
0000      01860      LD      A,49
0000      01870      RST      40
0000      01880      ENDM
0000      01890 @@VRSEC MACRO
0000      01900      LD      A,50
0000      01910      RST      40
0000      01920      ENDM

```

00000	01930	@@HDFMT	MACRO	
00000	01940		LD	A,52
00000	01950		RST	40
00000	01960		ENDM	
00000	01970	@@WRSEC	MACRO	
00000	01980		LD	A,53
00000	01990		RST	40
00000	02000		ENDM	
00000	02010	@@WRSSC	MACRO	
00000	02020		LD	A,54
00000	02030		RST	40
00000	02040		ENDM	
00000	02050	@@WRTRK	MACRO	
00000	02060		LD	A,55
00000	02070		RST	40
00000	02080		ENDM	
00000	02090	@@RENAM	MACRO	
00000	02100		LD	A,56
00000	02110		RST	40
00000	02120		ENDM	
00000	02130	@@REMOV	MACRO	
00000	02140		LD	A,57
00000	02150		RST	40
00000	02160		ENDM	
00000	02170	@@INIT	MACRO	
00000	02180		LD	A,58
00000	02190		RST	40
00000	02200		ENDM	
00000	02210	@@OPEN	MACRO	
00000	02220		LD	A,59
00000	02230		RST	40
00000	02240		ENDM	
00000	02250	@@CLOSE	MACRO	
00000	02260		LD	A,60
00000	02270		RST	40
00000	02280		ENDM	
00000	02290	@@BKSP	MACRO	
00000	02300		LD	A,61
00000	02310		RST	40
00000	02320		ENDM	
00000	02330	@@CKEOF	MACRO	
00000	02340		LD	A,62
00000	02350		RST	40
00000	02360		ENDM	
00000	02370	@@LOC	MACRO	
00000	02380		LD	A,63
00000	02390		RST	40
00000	02400		ENDM	
00000	02410	@@LOF	MACRO	
00000	02420		LD	A,64
00000	02430		RST	40
00000	02440		ENDM	
00000	02450	@@PEOF	MACRO	
00000	02460		LD	A,65
00000	02470		RST	40
00000	02480		ENDM	
00000	02490	@@POSN	MACRO	
00000	02500		LD	A,66
00000	02510		RST	40
00000	02520		ENDM	
00000	02530	@@READ	MACRO	

0000	02540	LD	A,67
0000	02550	RST	40
0000	02560	ENDM	
0000	02570 @@REW	MACRO	
0000	02580	LD	A,68
0000	02590	RST	40
0000	02600	ENDM	
0000	02610 @@RREAD	MACRO	
0000	02620	LD	A,69
0000	02630	RST	40
0000	02640	ENDM	
0000	02650 @@RWRTIT	MACRO	
0000	02660	LD	A,70
0000	02670	RST	40
0000	02680	ENDM	
0000	02690 @@SEEKSC	MACRO	
0000	02700	LD	A,71
0000	02710	RST	40
0000	02720	ENDM	
0000	02730 @@SKIP	MACRO	
0000	02740	LD	A,72
0000	02750	RST	40
0000	02760	ENDM	
0000	02770 @@VER	MACRO	
0000	02780	LD	A,73
0000	02790	RST	40
0000	02800	ENDM	
0000	02810 @@WEOF	MACRO	
0000	02820	LD	A,74
0000	02830	RST	40
0000	02840	ENDM	
0000	02850 @@WRITE	MACRO	
0000	02860	LD	A,75
0000	02870	RST	40
0000	02880	ENDM	
0000	02890 @@LOAD	MACRO	
0000	02900	LD	A,76
0000	02910	RST	40
0000	02920	ENDM	
0000	02930 @@RUN	MACRO	
0000	02940	LD	A,77
0000	02950	RST	40
0000	02960	ENDM	
0000	02970 @@FSPEC	MACRO	
0000	02980	LD	A,78
0000	02990	RST	40
0000	03000	ENDM	
0000	03010 @@FEXT	MACRO	
0000	03020	LD	A,79
0000	03030	RST	40
0000	03040	ENDM	
0000	03050 @@FNAME	MACRO	
0000	03060	LD	A,80
0000	03070	RST	40
0000	03080	ENDM	
0000	03090 @@GTDCT	MACRO	
0000	03100	LD	A,81
0000	03110	RST	40
0000	03120	ENDM	
0000	03130 @@GTDCB	MACRO	
0000	03140	LD	A,82

0000	03150	RST	40
0000	03160	ENDM	
0000	03170 @@GTMOD	MACRO	
0000	03180	LD	A,83
0000	03190	RST	40
0000	03200	ENDM	
0000	03210 @@RDSSC	MACRO	
0000	03220	LD	A,85
0000	03230	RST	40
0000	03240	ENDM	
0000	03250 @@GATRD	MACRO	
0000	03260	LD	A,86
0000	03270	RST	40
0000	03280	ENDM	
0000	03290 @@DIRRD	MACRO	
0000	03300	LD	A,87
0000	03310	RST	40
0000	03320	ENDM	
0000	03330 @@DIRWR	MACRO	
0000	03340	LD	A,88
0000	03350	RST	40
0000	03360	ENDM	
0000	03370 @@GATWR	MACRO	
0000	03380	LD	A,89
0000	03390	RST	40
0000	03400	ENDM	
0000	03410 @@MUL8	MACRO	
0000	03420	LD	A,90
0000	03430	RST	40
0000	03440	ENDM	
0000	03450 @@MUL16	MACRO	
0000	03460	LD	A,91
0000	03470	RST	40
0000	03480	ENDM	
0000	03490 @@DIV8	MACRO	
0000	03500	LD	A,93
0000	03510	RST	40
0000	03520	ENDM	
0000	03530 @@DIV16	MACRO	
0000	03540	LD	A,94
0000	03550	RST	40
0000	03560	ENDM	
0000	03570 @@DECHEX	MACRO	
0000	03580	LD	A,96
0000	03590	RST	40
0000	03600	ENDM	
0000	03610 @@HEXDEC	MACRO	
0000	03620	LD	A,97
0000	03630	RST	40
0000	03640	ENDM	
0000	03650 @@HEX8	MACRO	
0000	03660	LD	A,98
0000	03670	RST	40
0000	03680	ENDM	
0000	03690 @@HEX16	MACRO	
0000	03700	LD	A,99
0000	03710	RST	40
0000	03720	ENDM	
0000	03730 @@HIGH\$	MACRO	
0000	03740	LD	A,100
0000	03750	RST	40


```

0000      03760      ENDM
0000      03770 @@FLAGS MACRO
0000      03780      LD      A,101
0000      03790      RST      40
0000      03800      ENDM
0000      03810 @@BANK  MACRO
0000      03820      LD      A,102
0000      03830      RST      40
0000      03840      ENDM
0000      03850 @@BREAK MACRO #ADR
0000      03860      IFEQ     %,1
0000      03870      LD      HL,#ADR
0000      03880      ENDM
0000      03890      LD      A,103
0000      03900      RST      40
0000      03910      ENDM
0000      03920 @@CLS   MACRO
0000      03930      LD      A,105
0000      03940      RST      40
0000      03950      ENDM
0000      03960 @@CKBRKC MACRO
0000      03970      LD      A,106
0000      03980      RST      40
0000      03990      ENDM
0000      04000 *LIST   ON
0000      04010      END
00000 Total errors

```

